

الجمهورية العربية السورية

جامعة دمشق

كلية الهندسة المعلوماتية

قسم هندسة البرمجيات ونظم المعلومات

مقاربة وب دلالي لاستخدام الخدمات في نظم المعلومات خدميّة التوجّه

Semantic Web Approach for Using Services in Service Oriented Information Systems

دراسة أعدّت لنيل درجة الماجستير في الهندسة المعلوماتيّة

إعداد

المهندس صلاح الدين مراد

إشراف

الدكتور المهندس عمّار خيربك

2012-2011

كلمة شكر

﴿سُبْحَانَكَ لاَ عِلْمَ لَنَا إِلاَّ مَا عَلَّمْتَنَا إِنَّكَ أَنتَ الْعَلِيمُ الْحَكِيمُ﴾

إلى الدكتور عمّار خيربك أتقدّم بالشكر لما قدّمه لي من مساعدة وتوجيه لإتمام وإنجاز هذا البحث.

إلى الشمس التي أنارت دروب النجاح في حياتي "والدتي"،

إلى المربي الفاضل الذي نسج دروب النجاح في حياتي "والدي"،

إلى شريكتي وعوني في دروب حياتي "زوجتي"،

وإلى الشعب السوري العظيم أهدي هذا العمل المتواضع...

الفهرس

الجزء الثاني: الوب الدلالي

الفصل الثالث: تمثيل المعرفة والأنطولوجية في الوب الدلالي

الفصل الرابع: خدمات الوب الدلالية

الجزء الثالث: تطوير نموذج نظم المعلومات ذات الخدمات الدلالية

الفصل الخامس: النموذج النظري والرياضي لنظم المعلومات خدمية التوجّه الدلالية

الفصل السادس: النموذج العملياتي

الفصل السابع: تطبيق عملي

المقدّمة

لعل أهم نواحي هندسة البرمجيات التي جرى التركيز على تطويرها وإيجاد منهجيات لاستخدامها والبحث في آليات تطبيقها خلال العقود الماضية هو مفهوم إعادة الإستخدام، وقد أدى ذلك إلى ظهور العديد من المنهجيات والآليات في مجال هندسة البرمجيات. ويمكن النظر إلى إعادة الاستخدام على أنها فعالية وليست غرض، وهي تعني إستخدام شئ ما مرة أخرى، أي جعل الشيء يقوم بمهمة مزدوجة. وتعتبر إعادة الإستخدام فكرة قديمة وجديدة معاً، فمنذ بدايات البرمجة أعاد المبرمجون إستخدام الأفكار والأغراض والتجريدات والإجرائيات بطرق غير ممنهجة في غالبيتها.

تنوعت مستويات إعادة الإستخدام إبتداءً من مستوى التحقيق البرمجي لأجزاء من الرّماز الذي يصار إلى إعادة استخدامه ضمن إجرائيات في المنتج البرمجي نفسه، إلى تقسيم الرّماز الخاص بالمنتج البرمجي في بنية متعدّدة الطبقات إضافةً إلى استخدام أجزاء برمجية مختلفة في طريقة البناء والتحقيق من منتجات برمجية أخرى. ساعد على ذلك التطور ظهور لغات برمجية تدعم هذا التوجه فيما تقدّمه من إمكانية استخدام مكونات برمجية مستقلة عن لغة البرمجة المستخدمة وعن نظام التشغيل، وهذا ما يسمى بالبناء المعتمد على المكوّنات البرمجية. وقد واجه هذا التوجه تحديات عديدة منها: العمومية في بناء المكونات، وتصنيف المكونات، ومستودع المكونات، والوصول إلى المكونات.

تطوّر مفهوم الإعتماد على المكوّنات إلى استخدام المكوّنات البرمجية عن بعد (Remote Component Using)، وليتكوّن المنتج البرمجي عندئذٍ من اجتماع عدة أجزاء برمجية مطوّرة محلياً تعمل على منصة عمل ما وتتفاعل وتتبادل البيانات بطريقة أو بأخرى مع مكوّنات أخرى مطوّرة في مكان آخر وتعمل على منصة عمل أخرى، وهو ما يسمى ببرمجيات التطبيقات الموزّعة التي يمكن تعريفها بأنها "مجموعة من المكوّنات المنفصلة فيزيائياً والتي تعمل مع بعضها البعض كنظام برمجي واحد" [BARNABY 2002]. إن تطوّر مفهوم الأنظمة الموزّعة قد يتيح لنا القول وبمزيد من التجريد بأن "النظام الموزّع يتكون من مجموعة من الخدمات" [Fujii 2006] حيث يتم إستبدال كلمة مكونة بالخدمة.

ساعد تطوّر تقنيات الإنترنت في السنوات الماضية على ظهور العديد من الخدمات المتطوّرة ومنها خدمات الوب (Web Services)، حيث ساهمت خدمات الوب في انتشار ما يعرف بالبنية خدمية التوجّه (Service Oriented Architecture) والتي من الممكن تعريفها بأنها "عبارة عن مجموعة من الخدمات التي تتصل مع بعضها البعض بهدف تبادل البيانات أو للإشتراك في تأمين نشاط أو وظيفة معينة " [Barry 2003]. ويرى الباحثون بأن ظهور خدمات الوب والبنى خدمية التوجّه

سيساعد على الوصول إلى مكوّنات قابلة للتركيب والتبديل (Plug Compatible Components) أكثر من أي وقت مضى، مما سيساهم في تخفيض التكلفة وزيادة مقدرة الأنظمة.

يمكن القول بأن خدمات الوب والبنى خدمية التوجّه سوف:

1- تضيف خيارات إضافية في مجال تقنيات المعلومات؛
2- تجعل أنظمة تقنيات المعلومات أكثر مرونة وأكثر قدرة على التجاوب مع المتغيّرات؛
3- تساعد على تخفيض تكلفة التطوير؛
4- تساعد على تخفيض تكلفة الصيانة.

قد يكون من الصعب أحياناً إيجاد الخدمة المطلوبة لبناء النظام خدمي التوجّه بشكل مباشر، في الوقت الذي يكون من الممكن إيجاد تركيبة من تفاعل مجموعة من الخدمات لتوفير الخدمة النهائية المرجوة، وهي ما يمكن أن نسميه بالخدمة الإفتراضية (Virtual Service). ولعل أهم التحديات التي قد تواجهنا في حل مشاكل من هذا النوع هو أن تكون الخدمات المتوافرة مفهومة بالنسبة للآلة، وذلك حتى يتمكن النظام من إيجادها، ويحدد طريقة التخاطب معها، وبالتالي تقديم الخدمة النهائية المرجوة، ومن هنا ندرك أهمية ظهور الوب الدلالي كتطوّر مهم في عالم تقنيات الوب كمفهوم يساعد على التغلب على هذا النوع من المشاكل. ويعتبر الوب الدلالي الجيل الجديد من الوب العادي، وفيه تكون المعلومات قابلة للمعالجة من قبل الآلة (Machine Processable)، وقابلة للفهم من قبل الآلة أيضاً (Machine Understandable) [Stollberg 2005].

سنقوم من خلال هذه الأطروحة بدراسة تقنيات خدمات الوب والوب الدلالي لاستخدامها في إيجاد مقاربة دلالية لاستخدام الخدمات في النظم خدمية التوجّه (Semantic Based Service Oriented Information Systems).

سنقوم بتقسيم الأطروحة إلى ثلاثة أجزاء وهي الجزء الأول: نظم المعلومات خدمية التوجّه، الجزء الثاني الوب الدلالي، الجزء الثالث تطوير نموذج أنطولوجية نظم المعلومات ذات الخدمات الدلالية.

الجزء الأول

نظم المعلومات خدمية التوجّه

تطوّر بنى التطبيقات ونظم المعلومات

Development of Applications' Architecture and Information Systems

مقدّمة:

تطوّرت نظم المعلومات على نحوٍ مترافق مع تطوّر تقنيات المعلومات والتطبيقات خلال العقدين الماضيين، وقد جرى تركيز الإهتمام مؤخراً على مفهومين اثنين هما مفهوم إعادة الإستخدام (reusability) ومفهوم التشغيل البيني (interoperability). أدى ذلك إلى ظهور العديد من المنهجيات والآليات في مجال هندسة البرمجيات ونظم المعلومات التي تعكس هذه المفاهيم وتأثيرها على بنى التطبيقات وبنى نظم المعلومات عموماً.

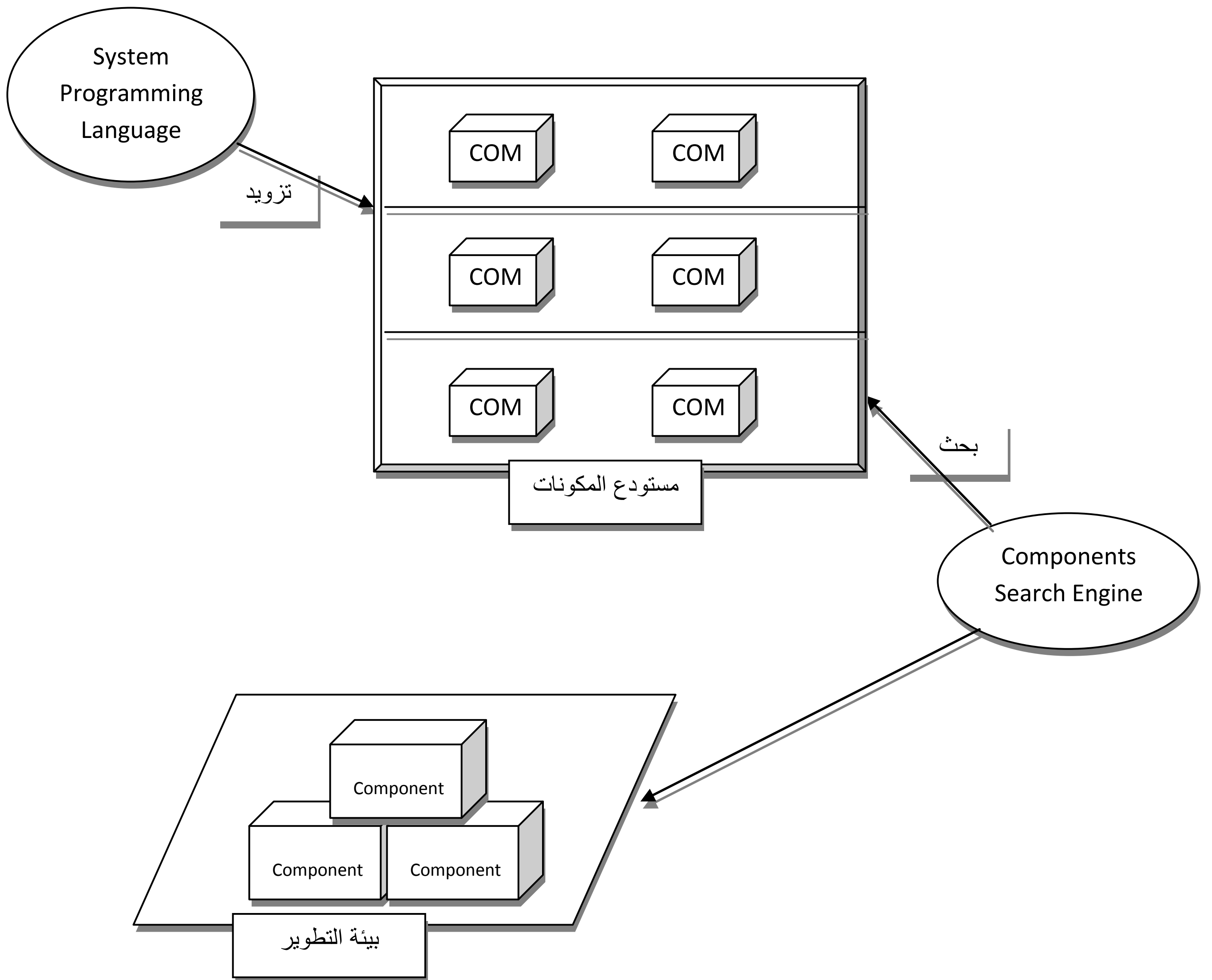
نستطيع تعريف إعادة الإستخدام على أنها إحتمال إستخدام شيء ما مرة أخرى لإضافة وظيفة جديدة دون الحاجة إلى إجراء تعديلات أو إجراء تعديلات طفيفة على هذا الشيء. تؤدي إعادة إستخدام الأشياء إلى التقليل من زمن التنفيذ وتقليل إمكانية الأخطاء والزمن اللازم للإستجابة للتغييرات [Parallab 2004]. كما يمكن تعريف التشغيل البيني على أنه قدرة أنظمة متعدّدة على العمل بعضها مع بعض مع مراعاة كافة العوامل التي تؤثر على وظيفة وأداء كل نظام على الآخر [Brien، Marakas 2007].

تعددّت بنى التطبيقات التي جرى تطويرها لتلبية متطلبات الأعمال فظهرت التطبيقات ثنائية وثلاثية الطبقات إلى التطبيقات متعدّدة الطبقات (n-tier applications). سنتقتصر الحديث في هذا الفصل عن البنى والمقاربات ذات الصلة بالمفهومين السابقين على نحو أكبر ومنها: مفهوم التطوير بالإعتماد على المكونات(Component Based)، والتطبيقات الموزّعة(Distributed Application)، والبنى خدمية التوجّه(Service Oriented Architecture).

1-1. التطبيقات المعتمدة على المكوّنات:

تعتمد منهجية التطوير المعتمد على المكونات (والتي تسمى أيضاً منهجية تجميع المكونات) على تجميع البرمجية من مكونات قابلة لإعادة الاستخدام، حيث تتواجد هذه المكونات بعدة أشكال في مستودعات (مكتبات) للمكونات (Components Repositories). حيث يتم تطوير المكونات عادةً– وليس دائماً - باستخدام لغات برمجة النظم مثل ++C، ومن ثم تضاف هذه المكونات إلى المكتبة، ويتم تجميع المكونات غالباً باستخدام لغات التغرية (Gluing languages). وتُستخدم محركات بحث خاصة للبحث عن المكونات ضمن مستودعاتها، حيث تعتمد هذه المحركات على عدة معايير للبحث والتطوير.

يوضح الشكل التالي "1-1" ملخص لمنهجية التطوير المعتمد على المكونات بأحدث أشكالها.

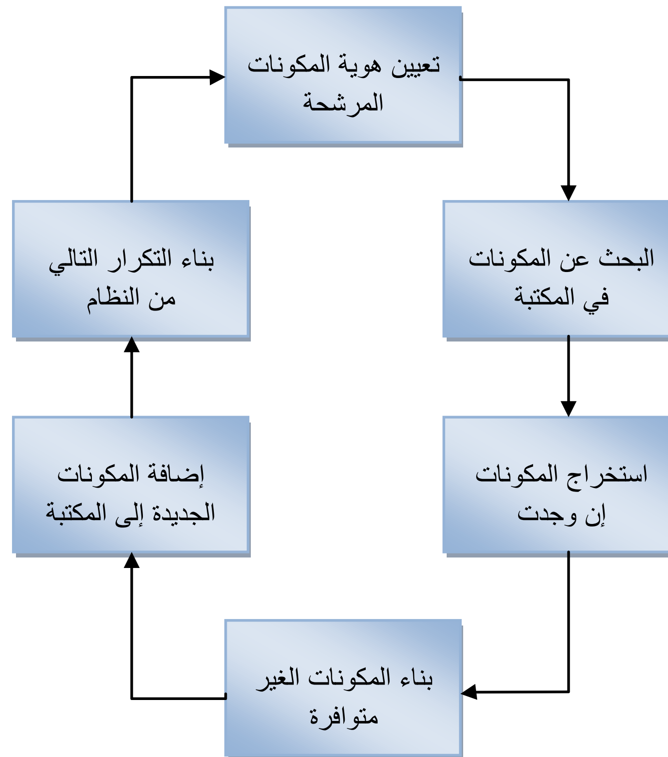


الشكل 1-1، ملخص لمنهجية التطوير المعتمد على المكونات

تختلف دورة حياة التطوير المعتمد على المكونات عن التطوير التقليدي بعدة طرق، فمرحلة تصميم نظام برمجي معتمد على المكونات تتضمن نشاطات جديدة كتحديد وإنشاء وتخصيص مجموعة من المكونات البرمجية، وأما في مرحلة التحقيق فيتم تحقيق التكامل بين هذه المكونات البرمجية مع بعضها داخل بنى برمجيات مناسبة. مما يتطلب وجود تقنيات تغرية تقوم بتغرية المكونات البرمجية القابلة لإعادة الاستخدام مع بعضها البعض لبناء النظام البرمجي بدلاً من كتابة النصوص البرمجية من الصفر [Barry 2008].

تبدأ عملية التطوير بتحليل النظام وتحديد الوظائف المطلوبة ومن ثم تحديد المكونات المرشحة للإستخدام، وبعدها يجري البحث عن هذه المكونات في مكتبة المكونات لمعرفة هل هذه المكونات موجودة سابقاً؟ وتستخلص من المكتبة إذا كانت موجودة فيها ويعاد إستخدامها. وفي حال عدم وجود مكونة ما في المكتبة تتم عندها هندسة هذه المكونة وتطور وتضاف إلى مكتبة المكونات كما يظهر الشكل التالي "2-1".

يشكل التزايد الأول من التطبيق المطلوب بناؤه باستخدام مكونات مستخلصة من المكتبة، وأية مكونات جديدة مبنية لتحقيق المتطلبات الخاصة بالتطبيق. وبنفس المنهجية تتشكل باقي التزايدات.



الشكل 2-1، خطوات التطوير بإعتماد على المكونات

لقد أفادت شركة QSM Associates إعتماداً على دراسة إعادة الإستخدام أن تجميع المكونات يقود إلى تقليل زمن دورة التطوير بنسبة 70% وتخفيض كلفة المشروع بنسبة 84%، ومع أنه هذه النتائج تابعة لقوة ومتانة مكتبة المكونات، فلا يوجد أدنى شك في أن منهجية تجميع المكونات تقدم ميزات هامة لهندسة البرمجيات في زمن الانترنت [Marakas 2007،Brien].

1-1-1. فوائد عملية التطوير المعتمد على المكونات:

لقد أشارت الدراسات التي أجرتها Gartner Group إلى أن الشركات ستقوم بتطوير البرمجيات بتغيير المنهجية القديمة لبناء التطبيقات- وهي منهجية البناء من الصفر- بمنهجية التطوير المعتمد على المكونات. وبالفعل لقد أصبحت منهجية التطوير المعتمد على المكونات أكثر استخداماً وذلك بسبب الفوائد التي تقدمها وهي:

1- تسرع منهجية التطوير المعتمد على المكونات من إنتاج التطبيقات:

إن تجميع التطبيقات من المكونات، ومن ثم كتابة النصوص البرمجية المطلوبة لتحقيق المتطلبات الجديدة فقط، أسرع من كتابة كامل التطبيق من الصفر [Microsoft 2006].

2- ينتج التطوير المعتمد على المكونات تطبيقات بنوعية عالية ووثوقية ممتازة:

بناء التطبيقات من مكونات موجودة يعني أن هذه المكونات قد استخدمت مراراً وهذا يعني أنها مختبرة جيداً- وهذا لا يغني بالطبع عن اختبار كامل التطبيق- وبالتالي فالتطبيقات المعتمدة على المكونات أكثر موثوقية من تلك المبنية من الصفر وذلك لأن المكونات مختبرة ومصححة على نحو أفضل [Barry 2008].

3- يجعل التطوير المعتمد على المكونات المطور يركز على مشاكل نطاق المسألة أكثر من التركيز على مشاكل عامة:

إن التطوير المعتمد على المكونات يجعل المطور يصرف جلّ وقته في حل مشاكل خاصة بنطاق تطبيقه ويريحه من تفاصيل برمجية عامة.

4- يخفض التطوير المعتمد على المكونات كلفة التطوير:

لأنه يبني تطبيقات موثوقة في وقت قصير، مما يخفض من تكاليف المبرمجين [Barry 2008].

5- يسمح التطوير المعتمد على المكونات بالمزج بين عدة لغات برمجة وباستخدام العديد من بيئات التطوير:

فالمكونات المكتوبة باستخدام لغة معينة- وهي غالباً من لغات برمجة النظم- يمكن وبسهولة إستخدامها من قبل لغة أخرى وهي غالباً لغة خطاطية (Scripting Languages).

6- يسهل التطوير المتعمد على المكونات صيانة التطبيق [Barry 2008]:

يمكن أن تبلغ صيانة البرمجيات الموجودة أكثر من 60% من مجمل الجهد الذي تبذله المؤسسة على التطوير، وهذا الأمر في ازدياد ويمكن مستقبلاً أن نرى مؤسسات لا تستطيع إنشاء تطبيقات جديدة لأنها غارقة في صيانة تطبيقاتها القديمة.

إن صيانة البرمجيات هي بالطبع أكثر من مشكلة "إصلاح إخطاء" بكثير. يمكننا تعريف الصيانة من خلال وصف أربع فعاليات يجري العمل بها بعد أن يصدر البرنامج للإستخدام [Brien، Marakas 2007]:

- الصيانة التصحيحية؛
- الصيانة التكيفية؛
- الصيانة التحسينية؛
- الصيانة الوقائية أو إعادة الهندسة.

يصرف 20% من جميع أعمال الصيانة في "إصلاح الأخطاء" فقط. وتصرف الـ80% المتبقية في تكييف النظم الموجودة، وإجراء التغييرات في بيئاته الخارجية، وإجراء التحسينات التي يطلبها المستخدم، وإعادة هندسة التطبيق للإستخدامات المستقبلية. عندما نقصد بالصيانة كل هذه الفعاليات يسهل علينا أن ندرك كيف أنها تتطلب الكثير من الجهد. لكن باعتماد منهجية التطوير المعتمدة على المكونات تصبح عملية الصيانة بكافة أنواعها أسهل وأقل هدراً للوقت، وذلك على اعتبار أن المكونات تشكل- كما في الهاردوير- قطع غيار للبرمجيات، وبالتالي يمكن إجراء تبديل لقطعة غيار (مكونة) في حال ظهور عيب فيها أو في حال عدم فعاليتها من حيث السرعة أو الوثوقية أو ماشابه.

1-1-2. صعوبات منهجية التطوير المعتمد على المكونات:

رغم كل ما سبق ذكره عن فوائد منهجية التطوير المعتمد على المكونات إلا أن هذه المنهجية كغيرها من المنهجيات تعاني من بعض العقبات أهمها:

1- صعوبة وكلفة تطوير المكونات البرمجية:

بما أن الهدف الرئيسي من المكونة البرمجية هو إعادة الإستخدام في أكثر من تطبيق، إذاً يجب بناء هذه المكونة بمنهجية عامة مع مراعاة كافة الحالات والهيئات التي قد تستخدم فيها هذه المكونة، وهذا سيؤدي إلى صعوبة أكبر في تطوير هذه المكونة، بالإضافة إلى الكلفة المادية والزمنية المرتفعة لعملية التطوير [Microsoft 2006]. لذلك يجب الإنتباه إلى هذه النقطة، ومعرفة أن الكلفة المرتفعة للمكونة ستوزع على جميع التطبيقات التي ستسخدم هذه المكونة، وليس فقط على التطبيق الذي طورت من أجله.

2- صعوبة تصنيف المكونات، وصعوبة البحث عنها:

يجب الانتباه إلى أن نجاح عملية التطوير المعتمدة على المكونات مرهون بوجود عدد كبير من المكونات البرمجية الجاهزة، وهذا بدوره سيؤدي إلى وجود مشكلة، وهي إيجاد منهجية للحصول على المكونة المطلوبة بأسرع وقت ممكن. لذلك تم تطوير عدة طرق لتصنيف المكونات واسترجاعها.

1-1-3. نجاح منهجية التطوير المعتمد على المكونات:

إن نجاح منهجية التطوير المعتمد على المكونات في بناء تطبيقات متطورة يتطلب إضافةً إلى استخدام المكونات ما يلي:

- إختيار بنية تطبيق فعالة؛
- إستخدام منهج التطوير التكراري.

ففي الحقيقة إن إنشاء بنية تطبيق صحيحة يبقى الجزء الأهم في أية عملية تطوير. ورغم أن المكونات تستطيع تحسين ذلك بعدة طرق، ولكن لأجل الأنظمة البرمجية الفعالة تبقى البنية البرمجية الجيدة ذات تأثير أكبر على نجاح عملية التطوير من توفر بعض المكونات. بالتأكيد إن امتلاك مقدار كبير من المكونات لنختار منها سيساعدنا في عملية التطوير ولكن ذلك غير كافٍ على الإطلاق.

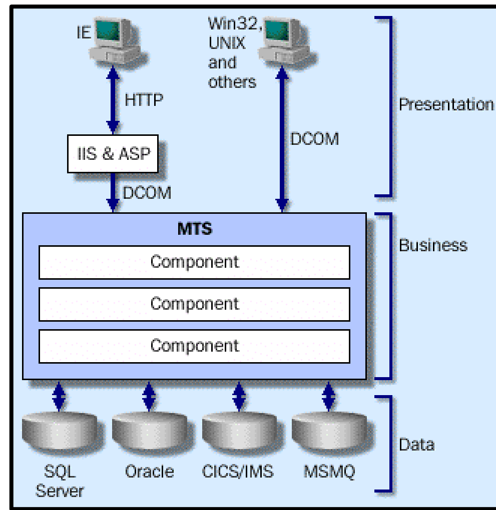
في الحقيقة إن قوة التطوير المعتمد على المكونات تأتي من التجميع الذكي للمكونات وبشكل هرمي، ومن إمكانية تكاملها مع بعضها وإعادة استخدامها، وليس من المكونة كمكونة مفردة. إن بناء المكونات وتجميعها يجعلنا نفكر بمنهجية مجردة، وبدون ذلك سنكون كمن يقوم بالتفكير بمنهجية عمل الترانزستور بدلاً من الـBoard و Chip.

إن إنشاء مكونات صحيحة وقابلة لإعادة الاستخدام يتطلب تصميم بنية صحيحة. أما مجرد تجميع الأنظمة المعقّدة من مستودعات المكونات، فإنه لن يعمل إلا مع الأنظمة البسيطة. أي أن تقنية المكونات مع إستخدام البنية الجيدة هما عمليتان مرتبطتان وضروريتان لنجاح منهجية التطوير المعتمد على المكونات.

2-1. التطبيقات الموزّعة:

يُعرّف التطبيق بشكل عام على أنه برنامج حاسب يتم استخدامه لحل مشكلة معينة أومجموعة من المشاكل المترابطة [Barry 2008]. يتم تشغيل التطبيقات البسيطة في فضاء عملية واحدة " single process". أما المخدمات فهي تطبيقات متخصصة تقوم بتزويد خدمات معينة إلى تطبيقات أخرى. ويطلق على التطبيق الذي يطلب الخدمة بالزبون. عادةً ما يتم تشغيل كل من المخدم والزبون على أجهزة منفصلة على الشبكة [Barry 2008]. ويمكن للمخدم أن يكون زبون في نفس الوقت حتى يكون قادر على تلبية احتياجات زبائنه. بينما تُعرّف التطبيقات الموزّعة على أنها التطبيقات التي يتم تنفيذها على أكثر من حاسب على الشبكة وهي عبارة عن مجموعة من التطبيقات المتخصصة التي تعمل معاً من أجل تحقيق هدف ما [Barry 2008].

يُظهر الشكل التالي "1-3" تطبيقاً موزعاً مع مجموعة من التقنيات التي من الممكن إستخدامها لتحققيه:



الشكل 1-3، تطبيق موزّع باستخدام تقنيات مختلفة

بالنسبة للمؤسسات فعلى هذه التطبيقات أن تكون مرنة اتجاه تغييرات الأعمال وتلبي احتياجات المستخدمين المتزايدة. ويمكن أن تنتشر على عدة منصات عمل على الشبكة الداخلية للمؤسسة، أو

حتى على الانترنت. ويجب أن تكون سهلة الاستخدام وتلبي احتياجات الأمن، والإدارة، والصيانة. باختصار يمكن القول بأنها أنظمة معقّدة.

1-2-1. فوائد التطبيقات الموزعة:

كباقي التطبيقات، على التطبيقات الموزّعة أن توفر واجهة إستخدام سهلة تلبي إحتياجات المستخدمين وغيرها من مواصفات الجودة العامة. إلى جانب هذه المتطلبات يمكن توصيف ثلاث متطلبات للتطبيقات الموزعة وهي:

1- قابلية التحمل (Scalable):

يجب أن لا يتأثر أداء التطبيق الموزع بشكل كبير نتيجة لازدياد عدد المستخدمين أو بزيادة ضغط العمل

2- الوثوقية (Reliable):

التطبيقات الموثوقة هي تلك التطبيقات التي لا تؤثر على إستمرارية عمل مستخدميها بسبب أي خلل برمجي أو عتادي. أيضاً يجب أن يكون للتطبيق مستوى عالٍ من الموثوقية لدى مستخدميه من جهة صحة عمله وتوافره بشكل دائم.

3- الكفاءة (Efficient):

على التطبيقات أن تتصرف بسرعة وفعالية من أجل الوصول بالمستخدمين إلى أهدافهم.

تتميز التطبيقات الموزعة على مستوى المؤسسات بمواصفات إضافية إلى جانب المواصفات السابقة وهي:

1- الشمولية (Extensive):

تتميز التطبيقات في المؤسسات بـ:

- تعدّد المستخدمين.
- تعدّد التطبيقات على آلات متعددة.
- تتعامل مع حجم هائل من البيانات.
- قد تستخدم المعالجة التفرعية.
- تستخدم موارد موزّعة على الشبكة.
- تعمل على منطق معقّد.

يتم تطوير هذه التطبيقات من قبل عدد من المطورين ويتم نشرها على منصات عمل متعددة وتتعامل مع العديد من التطبيقات الأخرى.

2- موجهة للأعمال(Business oriented):

تهدف إلى تلبية إحتياجات أعمال محددة في المؤسسة من سياسات، مهام، وقواعد أعمال.

3- عالية الأهمية (Mission critical):

على التطبيقات أن تتمتع بمتانة كبيرة يضمن إستمرارية عملها وتأدية وظيفتها. يجب أن تكون مرنة بشكل كافٍ للتكيف مع ازدياد ضغط العمل وازدياد عدد المستخدمين وتسمح بفعالية الصيانة، المراقبة، والإدارة.

1-2-2. الفوائد التكتيكية:

للتطبيقات الموزّعة عدة فوائد تكتيكية (على المدى القصير) وهي:

- تطبيق استراتيجية فرق تسد على تطوير التطبيقات؛
- إمكانية مركزية إدارة قواعد العمل؛
- إستخدام أكثر فاعلية للاتصال بقواعد البيانات؛
- مرونة أكبر بتوزيع البيانات (فالتطبيق الزبون للخدمة لا يعلم مكان البيانات)؛
- تسخير أكبر لإمكانيات المخدم (حيث يمكن لأكثر من تطبيق زبون استخدام المخدم)؛
- تصميم أفضل لطبقة الأمان من خلال استخدام الأدوار؛
- إمكانية تخصيص الزبائن بشكل أكبر (حيث يمكن للتطبيق الزبون الذي يستخدم مستعرض الويب أو للتطبيق المكتبي استخدام نفس المكونات الوسيطة).

1-2-3. الفوائد الاستراتيجية:

للتطبيقات الموزّعة عدة فوائد استراتيجية (على المدى البعيد) وهي:

- زيادة كفاءة العمال من خلال حقائق وإفتراضات أفضل؛
- عمال أفضل من خلال التفويض؛
- معرف جديدة كالتنجيم في البيانات (data mining).

1-3. البنى المعتمدة على الخدمات:

إن العامل الأساسي الذي يساعد المنظمات على المنافسة والنمو هو قدرتها على التغير السريع وتحسين إجراءات العمل. ويمكن للمنظمة أن تتأثر بشكل سلبي عند وجود بطء في تجاوب قسم تقنيات المعلومات العامل لديها مع تغيير متطلبات حاجات العمل [Microsoft 2006]. وتقوم الأعمال على ربط الأشخاص، والإجراءات، والمعلومات مع بعضها البعض ضمن نفس المؤسسة ومع المؤسسات التابعة والشركاء. وتظهر التكلفة الإضافية لذلك بشكل متزايد عند عدم وجود المرونة الكافية لربط الأنظمة، والتطبيقات، والبيانات، وغيرها من أصول تقنيات المعلومات. إن عدم المرونة هذه هو ما يؤدي إلى فشل تقنيات المعلومات بالإستجابة السريعة لتغير متطلبات الأعمال. باختصار، إن نقص التكامل والترابط يعد أحد أكبر التحديات التي تواجه المؤسسات في المنافسة والنمو [Microsoft 2006].

تُعرّف البنى المعتمدة على الخدمات بأنها مقاربة تصميمية لتنظيم أصول تقنيات المعلومات (من أنظمة وتطبيقات موزّعة ومعقّدة) في شبكة متكاملة، ومبسطة، وعالية المرونة [Microsoft 2006].

إن البنى الخدمية التوجة المنفّذة بشكل جيد تقوم بالمواءمة المباشرة بين موارد تقنيات المعلومات وأهداف الأعمال، مما يساعد المنظمة أو المؤسسة على بناء صلات أكثر قوةً مع زبائنها ومورديها، من خلال تأمين معلومات متكاملة، وصحيحة، وقابلة للقراءة بشكل أكبر بالتالي إمكانية اتخاذ قرارات

أكثر ذكاءً، ويساعد ذلك على تحسين أداء الموظفين من خلال المشاركة بهذه المعلومات. والنتيجة النهائية لهذا هي زيادة رشاقة المنظمة.

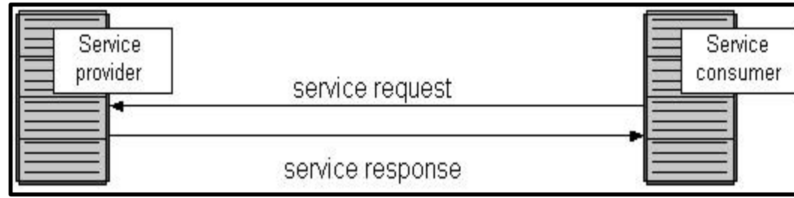
يمكن النظر إلى بنى الأنظمة المعتمددة على الخدمات أو الخدمية التوجه بأنها عبارة عن بنى تتألف بشكل أساسي من مجموعة من الخدمات التي تتصل مع بعضها البعض. قد يشمل هذا الإتصال المعالجة البسيطة للبيانات أو التنسيق بين مجموعة من الخدمات لتأدية نشاط ما [Barry 2008].

- الخدمة:

وهي عبارة عن وظيفة جيدة التعريف، تحتوي ذاتها، ولا تعتمد على سياق أو حالة خدمة أخرى [Barry 2008].

- الاتصال:

إن تقنية خدمات الويب هي تقنية الإتصال الأكثر شيوعاً في البنى خدمية التوجه. والتي تستخدم بدورها تقنية الـXML لإنشاء الاتصال [Barry 2008]. حيث يتم تعريف الإتصال الخاص بالطلب والإستجابات الناتجة عن هذا الطلب بطريقة متعارف عليها فيما بين كل من طالب الخدمة(Service Provider) ومستهلك الخدمة(Service Consumer) كما في الشكل التالي"1-4".



الشكل 1-4، الإتصال بين مزوّد ومستهلك الخدمة

تعتبر البنى خدمية التوجه عبارة عن بنى ذات أسلوب يهدف إلى إلغاء الإعتمادية بين الوكلاء البرمجيين. حيث تعتبر الخدمة كوحدة عمل يتم تأديتها من قبل مزوّد خدمة لإتمام نتيجة مرجوة من قبل مستهلك الخدمة [Barry 2008]. إن هذا المفهوم مجرد لحد كبير وبناءً عليه فإن البنى خدمية التوجه موجودة في كل شيء.

1-3-1. أمثلة عن البنى خدمية التوجه:

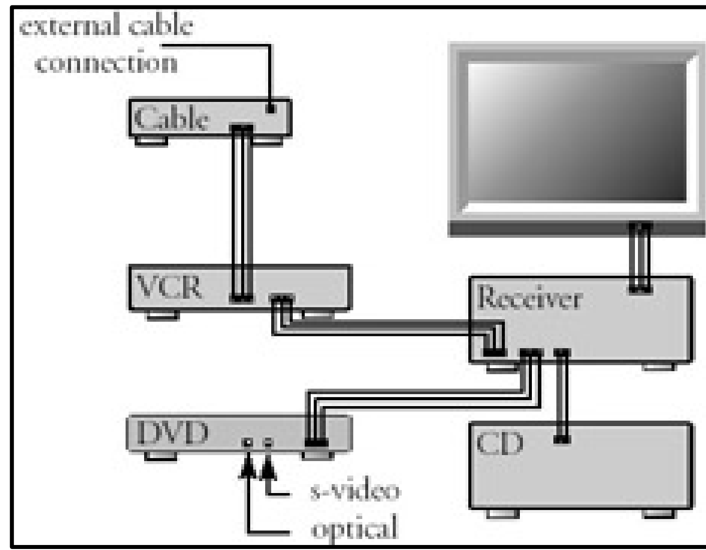
مثال 1 (مشغّل الأقراص المدمجة):

لننظر إلى المثال التالي الموجود لدى الكثيرين منا:

يتم استخدام القرص المدمج(CD) لتشغيل الموسيقى مثلاً من خلال وضعه ضمن مشغّل الأقراص المدمجة(CD Player) والذي يقوم بتشغيله. أي أن مشغل الأقراص المدمجة يقدم خدمة تشغيل الأقراص المدمجة. ويمكنك إستبدال مشغل الأقراص هذا بمشغل أقراص آخر بكل بساطة. كما ويمكنك تشغيل نفس القرص المدمج على عدة أنواع من مشغلات الأقراص المدمجة من مشغلات محمولة إلى مشغلات ستيريو، حيث يقدمان نفس الخدمة وهي تشغيل الأقراص المدمجة، ولكن جودة الخدمة مختلفة بينهما.

مثال 2 (خدمة العرض):

في مثال آخر نلاحظ فيما يلي اجتماع مجموعة من مزودي الخدمات التي يمكن لكل منها تأدية وظيفة ما بالتعاون فيما بينها. فمشغل الأُقراص المدمجة (CD) يقوم بالعرض على الشاشة باستخدام المستقبل (Receiver) من خلال مجموعة من الوصلات (Connections) حيث يمكن لمشغل الأقراص القيام بوظيفته مهما كان نوع الشاشة ويمكن تغيير الشاشة بأي وقت دون أن يؤثر ذلك على قدرة مشغل الأقراص على القيام بوظيفته. كذلك في حال وجود عطل في المستقبل فيتم تبديله بآخر دون التأثير على وظائف المكونات الأخرى. المهم هنا إمكانية وجود نوع الإتصال المناسب الذي يمكن لكلا الطرفين التعامل معه كما في الشكل التالي "1-5":



الشكل 1-5، خدمة عرض وتشغيل الأقراص المدمجة

مثال 3 (خدمة الطاقة الكهربائية):

حتى نكون فكرة أساسية وواضحة عن الأساليب ذات التوجه الخدمي بمثال آخر نأخذ طريقة تعاملنا مع الكهرباء، أنظر الشكل التالي، فكوننا مستخدمين لطاقة الكهرباء كونها في المنزل أوالعمل يكون لدينا إهتمام فقط في شكل القابس الكهربائي وجهد الكهربائي (110،220) فإذا احترمنا هاذين الأمرين أصبح بإمكاننا تشغيل كل الأجهزة الكهربائية لدينا بكل سهولة دون أن نعرف من أين مصدر هذه الطاقة كونها شمسية، نووية أوغازية، فكوننا مستخدمين أو مستهلكين لهذه الطاقة يحب علينا فقط معرفة واحترام شكل القابس الكهربائي وجهد الكهربائي (110،220).

كذلك أيضاً عند منتجي الطاقة فليس لديهم نية للتعرف على كل الأجهزة المنزلية أو أي أجهزة كهربائية مستقبلية جديدة، فقط يريدوا أن يعرفوا ما هوالجهد الكهربائي المطلوب (110،220) لتوفيره للمستخدمين، إذاً بهذا التصور أصبح من السهل تشبيه الأساليب ذات التوجه الخدمي بأنها ذلك القابس الذي يربط بين المنتجين بالمستخدمين من خلال تعريف واحترام بعض القواعد والمعلومات.

من الشكل التالي نستطيع إعادة تعريف الأساليب ذات التوجه الخدمي بأنها هي تلك البنية التحتية الخدمية التي تقوم ببناء أنظمة مرنة تتكيف بسرعة لظروف العمل المتغيرة لدعم بيئة الأعمال وتكنولوجيا المعلومات.

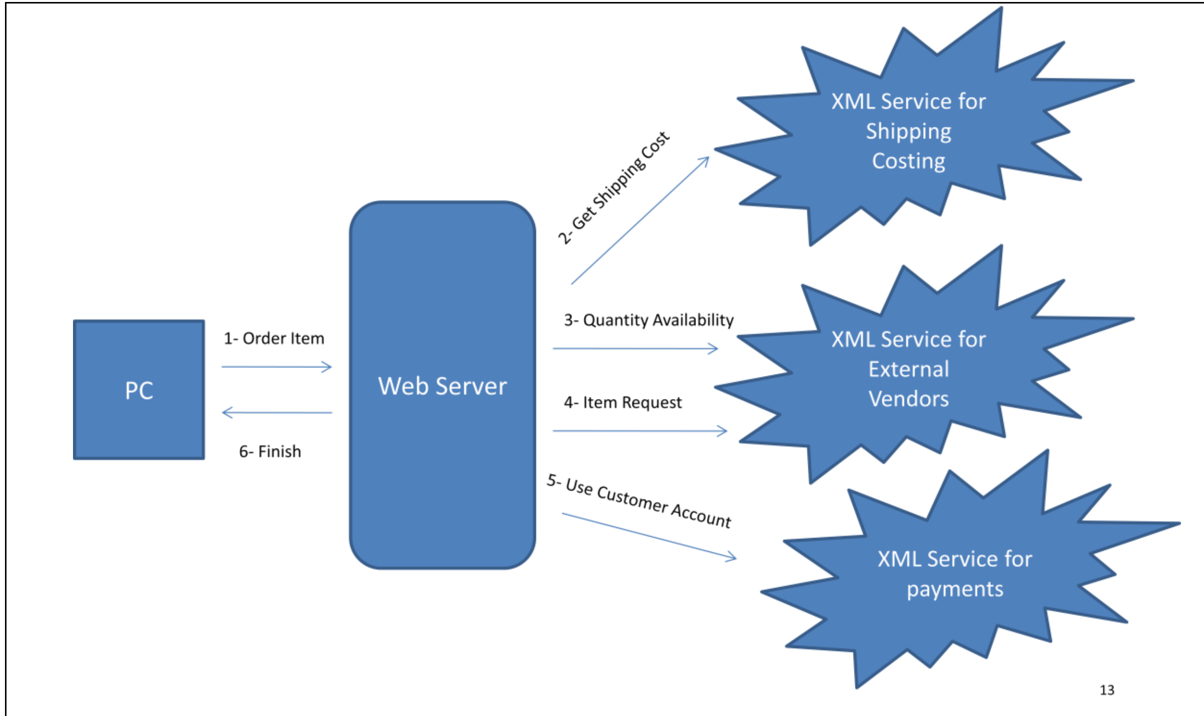


الشكل 1-6، خدمة الطاقة الكهربائية

مثال 4 (خدمة الشراء الإلكتروني):

هذا المثال يوضح نظام معلومات ذو بنية خدمية التوجه، يعتمد هذا النظام في بنيته على استخدام مجموعة من خدمات الويب المتواجدة عبر الإنترنت كخدمة الدفع الالكتروني، وخدمة شحن البضائع وغيرها من الخدمات لتشكيل ما يمكن اعتباره نظام المخزن الإفتراضي (Virtual Warehouse).

حيث يتم نشر و تعميم واجهة التخاطب مع هذه الخدمات (Service Interface) بشكل قابل للإستكشاف، بحيث توضح هذه الواجهة طريقة التفاعل مع الخدمة المقدمة وتكون بمثابة العقد (Contract) بين مزود الخدمة و مستهلكها كما في الشكل التالي "1-7".



الشكل 1-7، خدمة الشراء الالكتروني

1-3-2. مبادئ الأساليب ذات التوجه الخدمي الأساسية:

1- إعادة الإستخدام (Reuse):

إمكانية استخدام هذه الوحدات في عمليات مختلفة وأن تكون متوفرة عند الطلب.

2- الاستقلالية (Autonomous):

عدم الإعتمادية في استخدام هذه الوحدات على بعضها البعض عند تنفيذ العمليات.

3- قابلية التشغيل البيني (Interoperability):

القدرة على الإتصال، وتنفيذ البرامج، أولنقل البيانات بين مختلف وحدات تقنية المعلومات في نحو لا يتطلب من المستخدم أن يكون لديه معرفة مسبقة في خصائص هذه الوحدات.

4- قابلية التركيب (Composability):

القدرة على تأليف وتركيب الخدمات.

5- Granularity:

العمل على أصغر مستوى بحيث إمكانية التعامل في أدنى مستوى للخدمة.

6- قابلية النقل (Portability):

القدرة على الإنتقال من نظام إلى نظام أخر من دون أي تكاليف.

7- الوحدات الجزئية (Modularity):

التعامل على شكل وحدات منفصلة وقدرة فصل الخدمات كل على حدا.

8- الإذعان للمعايير (Standard compliance):

الإلتزام بالمعايير والمواصفات القياسية.

9- تحديد الخدمات (Service identification):

القدرة على تحديد الخدمات.

10- التصنيف (Categorization):

القدرة على تصنيف وتبويب الخدمات.

11- التوصيل (Delivery):

القدرة على توصيل الخدمة إلى طالبيها.

12- المراقبة والمتابعة (Monitoring and Tracking):

القدرة المراقبة والمتابعة والتسجيل.

1-3-3. مبادئ الأساليب ذات التوجه الخدمي التصميمية:

1- تغليف الخدمات (Service encapsulation):

وهي تلك المقدرة على تغليف مجموعة من الأوامر والتعليمات التي يمكن نقلها عبر الشبكات والانترنت.

2- قلة الاعتمادية (Service loose coupling):

عبارة عن الوصلات المحررة التي تقوم على المحافظة على العلاقات وتقليل الاعتمادية بينهما وتعتمد على الإدراك لمعرفة الخدمات المتوفرة.

3- عقد الخدمة (Service contract):

وهوعقد الخدمة الذي يعرف طريقة الاتصال ويحتوى على التعريفات اللازمة للخدمة.

4- التجريد (Service abstraction):

هو تجريد الخدمة من أي معلومات تحوي كيفية عملها وتحتوي فقط على وصف الخدمة للعالم الخارجي.

5- الجودة (Service optimization):

تقديم خدمات ذات جودة عالية.

6- قابلية الاستكشاف (Service discoverability):

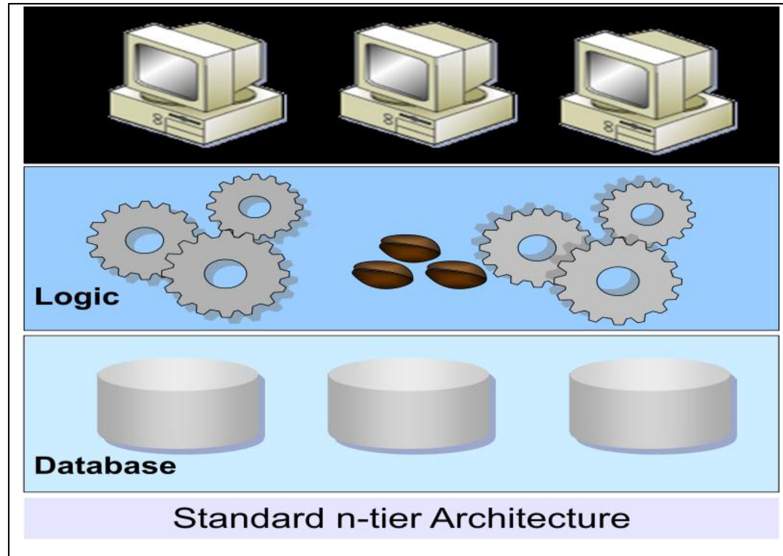
قدرة الخدمة على أن تكون قابلة للاكتشاف قبل آليات الكشف عن الخدمات.

7- إستقلالية الخدمة (Service autonomy):

هي إستقلالية الخدمة بحيث كل خدمة لديها القدرة على السيطرة على محتواها الخاص.

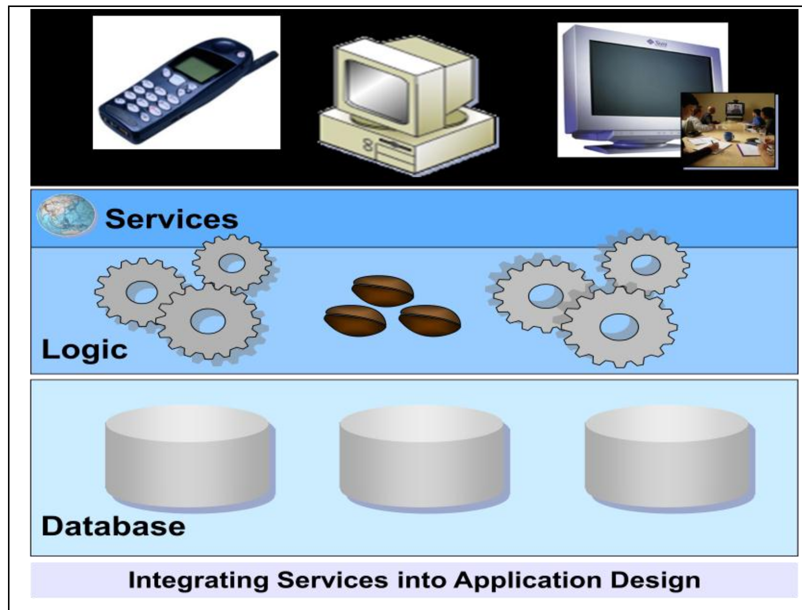
4-3-1. بنية التطبيقات في الأنظمة خدمية التوجه:

بالعودة إلى بنية التطبيقات متعددة الطبقات والتي نلاحظ فيها وجود ثلاث طبقات بشكل أساسي وهي طبقة العرض، طبقة منطق العمل، وطبقة قواعد البيانات كما في الشكل التالي "1-8":



الشكل 1-8، بنية التطبيق متعدد الطبقات

يتم في بنية الطبقات في الأنظمة خدمية التوجه وباستخدام خدمات الويب إضافة طبقة جديدة إلى البنية السابقة لعرض منطق التطبيق إلى العالم الخارجي على شكل مجموعة من الخدمات كما هو موضح في الشكل التالي "1-9":



الشكل 1-9، بنية تطبيقات خدمات الوب

1-3-5. أنواع الخدمات:

1- الخدمات عديمة الحالة (Stateless Service):

يتم في هذا النوع إرسال كامل المعلومات التي يحتاجها مزود الخدمة مع كل رسالة يقوم بإرسالها طالب أو مستهلك الخدمة. إن هذا يجعل مزود الخدمة قادر أكثر على الاستجابة للتغييرات(more scalable) لأنه لا يحتاج إلى حفظ معلومات عن حالة الخدمة بين الطلبات التي يستقبلها.

إن هذه الطريقة تُسهّل على برامج المراقبة متابعة الخدمات وتنفيذها بسهولة أكبر حيث سيهتم البرنامج بمراقبة طلب وحيد ومعرفة هدفه بدون القلق بشأن وجود حالة وسيطة معينة للخدمة قبل أو بعد هذا الطلب. كذلك فإن استعادة الخدمة بعد خلل ما سيكون أكثر سهولة والخدمة بشكل عام ستتمتع بوثوقية أعلى.

2- الخدمات المحتفظة للحالة (Stateful Service):

من الصعب في بعض الحالات تجنب استخدام هذا النوع من الخدمات. أحد هذه الحالات هو بدء جلسة (Session) بين مزود ومستهلك الخدمة. فعلى سبيل المثال في بعض الحالات يكون من المهم جداً لكل من مزود ومستهلك الخدمة إرسال شهادة أمن(Security certification) مع كل طلب. يمكن لتسريع العملية استبدال هذه الشهادة برمز مشترك(Shared token) فقط بين مزود ومستهلك الخدمة.

يتطلب هذا النوع من الخدمات من كل من مزود ومستهلك الخدمة التشارك في السياق المحدد نفسه لمستهلك الخدمة، والذي يمكن أن يكون موجوداً في الرسائل المتبادلة بين الطرفين أو تتم الإشارة إليه في هذه الرسائل.

إن من معوقات هذا القيد هي أنه سيقلل من قدرة مزود الخدمة الاستجابة للتغيرات(reduce scalability) لأنه سيكون ملزماً بحفظ معلومات عن السياق المشترك لكلٍ مستهلكي الخدمة لديه. كما يزيد ذلك من الاعتمادية(increase coupling) بين مزود ومستهلك الخدمة بحيث يكون تبديل مزود الخدمة غاية في الصعوبة.

1-4. خلاصة:

قمنا من خلال هذا الفصل بعرض تطور بنى التطبيقات لتلبية متطلبات الأعمال بحيث تم التركيز على المقاربات ذات الصلة بمفهومي إعادة الإستخدام والتشغيل البيني ومنها: مفهوم التطوير المعتمد على المكونات، والتطبيقات الموزّعة، والبنى خدمية التوجه. ولقد هدفنا من ذلك البحث في ظهور البنى خدمية التوجه حيث قمنا بتعريف أهم المشاكل التي واجهت التقنيات التي رافقت ظهور هذه البنى.

خدمات الوب والنظم خدمية التوجّه

Web Services and Service Oriented Architecture

مقدّمة:

ساعد تطوّر تقنيات الإنترنت في السنوات الماضية على ظهور العديد من الخدمات المتطورة ومنها خدمات الوب (Web Services)، حيث ساهمت خدمات الوب في انتشار ما يعرف بالبنية خدمية التوجه (Service Oriented Architecture) والتي من الممكن تعريفها بأنها "عبارة عن مجموعة من الخدمات التي تتصل مع بعضها البعض بهدف تبادل البيانات أوللإشتراك في تأمين نشاط أو وظيفة معينة " [Barry 2003].

يرى الباحثون بأن ظهور خدمات الوب والبنى خدمية التوجه سيساعد على الوصول إلى مكونات قابلة للتركيب والتبديل (Plug Compatible Components) أكثر من أي وقت مضى، مما سيساهم في تخفيض التكلفة وزيادة مقدرة الأنظمة. كما ويمكن القول بأن خدمات الوب والبنى خدمية التوجه سوف:

1- تضيف خيارات إضافية في مجال تقنيات المعلومات؛
2- تجعل أنظمة تقنيات المعلومات أكثر مرونة وأكثر قدرة على التجاوب مع المتغيرات؛
3- تساعد على تخفيض تكلفة التطوير؛
4- تساعد على تخفيض تكلفة الصيانة.

سنبحث في هذا الفصل ظهور خدمات الوب كتطور مهم في تقنيات الوب والذي ينتقل بالوب من الجانب الستاتيكي إلى الجانب الديناميكي، بالإضافة إلى أهمية إستخدام خدمات الوب في النظم خدمية التوجه.

1-2. خدمات الوب:

تعريف خدمات الوب:

خدمات الوب وهي نظم برمجية تم تصميمها بشكل يدعم قابلية التشغيل بالتفاعل بين أكثر من آلة على الشبكة. يتم ذلك من خلال واجهة توصيف الخدمة المكتوبة بلغة الـWSDL (Web Service Description Language). ويتم تبادل الرسائل مع الأنظمة الأخرى باستخدام بروتوكول SOAP (Simple Object Access Protocol) والذي يتم عبر بروتوكول الـHTTP [W3C].

تعريف الخدمة (Service):

من الممكن تعريف الخدمة في نظم المعلومات خدمية التوجه على أنها وظيفة (Function) جيدة التعريف، مستقلة، ولاتعتمد على سياق أوحالة خدمة أخرى [Barry 2003].

الإتصال (Connection):

يتم في البنى خدمية التوجه تعريف طريقة إتصال مفهومة من قبل كلٍ من مستهلك (Consumer) ومزود (Provider) الخدمة، ويتم استخدام طريقة الإتصال هذه أثناء الطلب (Request) والإستجابات (Responses) المترتبة على هذا الطلب، وتقع الخدمة في طرف قنوات الإتصال.

إن من الأمثلة الأكثر شيوعاً لخدمات الوب المنتشرة على الإنترنت: حالة الطقس، أسعار بضائع شركة ما، برامج محطة تلفزيونية، بالإضافة إلى خدمات أخرى مثل تخزين الملفات، الضرائب ..... الخ.

1-1-2. لغة توصيف خدمات الوب:

إن لغة وصف خدمات الوب WSDL (Web Services Description Language) هي عبارة عن لغة مبنية على ترميز XML لوصف خدمات الوب على الانترنت وتسهيل الوصول والتعامل معها كأن يقوم مثلا بتحديد موقع خدمة الوب وطرق وعمليات هذه الخدمة.

مميزات الـ WSDL:

- تستخدم لتوصيف خدمات الوب؛
- مكتوبة بلغة الـ XML؛
- عبارة عن مستند XML.

التطور التاريخي للغة WSDL ضمن W3C:

قُدّمت النسخة الأولى WSDL 1.1 من هذه اللغة كمذكرة W3C مقدمة من قبل مجموعة من الشركات لتوصيف الخدمات معتمدة على بروتوكولات XML. حيث أن مذكرات W3C هي عبارة عن مذكرات تقوم مؤسسة W3C بجعلها متاحة للجميع للنقاش حولها وتطويرها من قبل عدة شركات ومؤسسات بحث معا، ونشر المذكرات بواسطة W3C لا يعني ضرورة أن تكون قد كتبت من قبل

المؤسسة ذاتها أو بواسطة فريق المؤسسة أو أحد أعضائها. صدرت أولى النسخ للإصدار WSDL 1.2 من قبل W3C في تموز 2002.

مستند الـ WSDL:

يمكن القول أن مستند WSDL هو مستند بسيط مكتوب كمستند XML ويحتوي على مجموعة من التعاريف التي تعرف خدمة وب.

بنية مستند WSDL:

يعرّف مستند الـ WSDL خدمة وب ما باستخدام العناصر الرئيسية التالية:

العنصر(Element)	الوصف
<portType>	يقوم بتعريف العمليات التي تقوم خدمة وب ما بتنفيذها.
<message>	يقوم بتعريف الرسائل(Messages) المستخدمة من قبل خدمة وب ما.
<types>	يقوم بتعريف أنماط المعطيات(data types) المستخدمة من قبل خدمة وب ما.
<binding>	يقوم بتعريف بروتوكولات الاتصال المستخدمة من قبل خدمة وب ما.

وتبدوالبنية الرئيسية لمستند WSDL على النحوالتالي:

```
<definitions>
<types>
  definition of types........
</types>
<message>
  definition of a message....
</message>
<portType>
  definition of a port.......
</portType>
<binding>
  definition of a binding....
</binding>
</definitions>
```

يمكن لمستند WSDL أن يحتوي أيضاً عناصر أخرى مثل عناصر التوسعة(Extension Elements) مثلاً أوعنصر خدمة يجمع عدة تعريفات لخدمات وب متعددة معاً ضمن مستند WSDL مفرد.

منافذ الـ WSDL (WSDL Ports):

يعتبر المحدد <portType> أحد أهم المعرّفات ضمن مستند WSDL، يقوم هذا العنصر بتعريف خدمة الوب، والعمليات التي يمكن تنفيذها، والرسائل التي يمكن استخدامها. يمكن أن يتم اعتبار العنصر <portType> كمكتبة أو صف أو وحدة ضمن لغات البرمجة التقليدية.

رسائل الـ WSDL (WSDL Messages):

يقوم العنصر <message> بتعريف الرسائل المستخدمة من قبل خدمة وب ما، حيث يقوم بتعريف عناصر المعطيات لعملية ما. يمكن لرسالة ما أن تتألف من جزء واحد أو أكثر، يمكن تشبيه أجزاء الرسالة كبارامترات تابع ما ضمن لغات البرمجة التقليدية.

أنماط WSDL (WSDL Types):

يقوم العنصر <types> بتعريف أنماط المعطيات(data types) المستخدمة من قبل خدمة وب ما، ومن أجل أمور التوافقية مع كل اللغات الأخرى تستخدم لغة WSDL نفس قواعد "XML Schema" لتعريف أنماط المعطيات.

روابط WSDL:

يقوم العنصر <binding> بتعريف بروتوكولات الإتصال المُستخدمة من قبل خدمة وب ما، كما يقوم بتعريف صيغة الرسالة ويقوم بذلك من أجل كل منفذ.

مثال لمستند WSDL:

فيما يلي مثال يوضح جزء من مستند WSDL:

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>
<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>
<portType name="glossaryTerms">
  <operation name="getTerm">
     <input message="getTermRequest"/>
     <output message="getTermResponse"/>
  </operation>
</portType>
```

في المثال السابق يقوم العنصر "portType" بتعريف "glossaryTerms" كإسم للمنفذ كما يعرف عملية ضمنه بإسم "getTerm"، كما يتم تعريف العملية "getTerm" التي لها رسالة دخل

بالاسم "getTermRequest" ورسالة خرج بالاسم"getTermResponse". تعرف عناصر الرسالة message أجزاء كل رسالة وانماط المعطيات الموافقة لها. بالمقارنة مع لغات البرمجة التقليدية يعتبر المنفذ "glossaryTerms" كمكتبة و"getTerm" تابع ضمن المكتبة له متحول دخل هو"getTermRequest"ومتحول خرج هو"getTermRequest".

منافذ الـ WSDL:

يقوم منفذ الـ WSDL بوصف الواجهات (العمليات القانونية) التي سيتم استدعاؤها من قبل خدمة وب، كما ذكرنا سابقاً فإن العنصر <portType> هو أكثر عناصر لغة WSDL أهمية حيث يقوم هذا العنصر بتعريف خدمة الوب والعمليات التي يمكن إنجازها والرسائل المتبادلة. يعرف المنفذ نقطة اتصال لخدمة الوب ويمكن تشبيهه بمكتبة أو صف أو وحدة في لغة برمجة تقليدية وكما ذكرنا سابقاً يمكن تشبيه كل عملية ضمنه بالتابع ضمن لغة برمجة تقليدية.

أنواع العمليات :

يعتبر النوع طلب-استجابة "request-response" النوع الأكثر شيوعاً واستخداماً ولكن مع ذلك تعرف لغة WSDL الأنماط الأربعة التالية:

النوع	التعريف
One-way	اتجاه واحد : يمكن للعملية استقبال رسالة ولكن لن تعيد استجابة.
Request-response	طلب_ استجابة: يمكن للعملية استقبال رسالة وسوف تعيد استجابة.
Solicit-response	استجداء – استجابة : يمكن للعملية إرسال طلب ومن ثم تقوم بالانتظار لاستجابة.
Notification	إشعار: يمكن للعملية إرسال رسالة وسوف لن تقوم بالانتظار لاستجابة.

- عملية طلب باتجاه واحد "One Way":

فيما يلي مثال على عملية ذات اتجاه واحد:

```
<message name="newTermValues">
  <part name="term" type="xs:string"/>
  <part name="value" type="xs:string"/>
</message>
<portType name="glossaryTerms">
  <operation name="setTerm">
    <input name="newTerm" message="newTermValues"/>
  </operation>
</portType >
```

في المثال السابق يقوم المنفذ "glossaryTerms" بتعريف عملية باتجاه واحد تدعى "setTerm". تسمح هذه العملية بإدخال مصطلحات جديدة للقاموس باستخدام رسالة "newTermValues" التي تأخذ بارامترات دخل هي "term" و"value" ولا يتم تعريف أي خرج لهذه العملية.

- عملية طلب- استجابة "Request-Response":

فيما يلي مثال عن هذا النوع من العمليات:

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>
<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>
<portType name="glossaryTerms">
 <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
 </operation>
</portType>
```

في هذا المثال يقوم المنفذ "glossaryTerms" بتعريف عملية من النوع طلب – استجابة وهي "getTerm"، تقوم هذه العملية بإيجاد القيمة المقابلة لمصطلح ما تتطلب هذه العملية رسالة دخل تدعى "getTermRequest" والتي لها المتحول "term" وسوف تعيد لنا رسالة خرج "getTermResponse" مع متحول يدعى "value".

روابط الـ WSDL :

كما ذكرنا يقوم العنصر <binding> بتعريف بروتوكولات الإتصال المُستخدمة من قبل خدمة وب ما كما يقوم بتعريف صيغة الرسالة ويقوم بذلك من أجل كل منفذ.

الربط إلى SOAP (Binding to SOAP):

لنأخذ المثال التالي الذي يحتوي عملية طلب – استجابة.

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>
<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>
<portType name="glossaryTerms">
 <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
```

```
  </operation>
</portType>
<binding type="glossaryTerms" name="b1">
<soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http" />
  <operation>
    <soap:operation
     soapAction="http://example.com/getTerm"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
```

يمتلك العنصر binding واصفتين هما واصفة الاسم name، وواصفة النوع type. تعرف واصفة الإسم name إسم عملية الربط، أما واصفة النوع type فتشير إلى المنفذ المطلوب لعملية الربط. طبعاً في حالتنا المنفذ "glossaryTerms". يمتلك العنصر soap:binding واصفتين أيضاً: واصفة التصميم style، وواصفة النقل transport. يمكن أن تكون واصفة التصميم style من الشكل "rpc" أومن الشكل "document"، أما واصفة النقل HTTP فتقوم بتعريف البروتوكول المستخدم مع SOAP. يقوم العنصر operation بتعريف كل عملية يعرضها المنفذ.

قواعد WSDL (WSDL Syntax):

فيما يلي القواعد الكاملة للإصدار WSDL Syntax كما تم توصيفه في الورقة الصادرة عن W3C:

```
<wsdl:definitions name="nmtoken"? targetNamespace="uri">
    <import namespace="uri" location="uri"/> *
    <wsdl:documentation .... /> ?
    <wsdl:types> ?
        <wsdl:documentation .... /> ?
        <xsd:schema .... /> *
    </wsdl:types>
    <wsdl:message name="ncname"> *
        <wsdl:documentation .... /> ?
        <part name="ncname" element="qname"? type="qname"?/> *
    </wsdl:message>
    <wsdl:portType name="ncname"> *
        <wsdl:documentation .... /> ?
        <wsdl:operation name="ncname"> *
            <wsdl:documentation .... /> ?
            <wsdl:input message="qname"> ?
```

```
          <wsdl:documentation .... /> ?
        </wsdl:input>
        <wsdl:output message="qname"> ?
          <wsdl:documentation .... /> ?
        </wsdl:output>
        <wsdl:fault name="ncname" message="qname"> *
          <wsdl:documentation .... /> ?
        </wsdl:fault>
      </wsdl:operation>
    </wsdl:portType>
    <wsdl:serviceType name="ncname"> *
      <wsdl:portType name="qname"/> +
    </wsdl:serviceType>
    <wsdl:binding name="ncname" type="qname"> *
      <wsdl:documentation .... /> ?
      <-- binding details --> *
      <wsdl:operation name="ncname"> *
        <wsdl:documentation .... /> ?
        <-- binding details --> *
        <wsdl:input> ?
          <wsdl:documentation .... /> ?
          <-- binding details -->
        </wsdl:input>
        <wsdl:output> ?
          <wsdl:documentation .... /> ?
          <-- binding details --> *
        </wsdl:output>
        <wsdl:fault name="ncname"> *
          <wsdl:documentation .... /> ?
          <-- binding details --> *
        </wsdl:fault>
      </wsdl:operation>
    </wsdl:binding>
    <wsdl:service name="ncname" serviceType="qname"> *
      <wsdl:documentation .... /> ?
      <wsdl:port name="ncname" binding="qname"> *
        <wsdl:documentation .... /> ?
        <-- address details -->
      </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

2-1-2. البروتوكول SOAP:

وهو بروتوكول بسيط يعتمد على الـXML يسمح للتطبيقات بتبادل المعلومات فيما بينها عبر بروتوكول الـHTTP، كما يُعرّف بأنه بروتوكول للوصول إلى خدمات الوب [W3C 2001]. الـSOAP:

- هو اختصار لـSimple Object Access Protocol؛
- هو بروتوكول اتصال(Communication protocol)؛
- يستخدم للإتصال بين التطبيقات؛
- هو صيغة لإرسال الرسائل؛
- يتصل عبر الانترنت؛
- مستقل عن البيئة؛
- مستقل عن اللغة؛
- يعتمد على الـXML؛
- بسيط وقابل للتوسيع؛
- يمكنه تجاوز جدران النار (Firewalls).

إن الطريقة الأخرى للتواصل بين التطبيقات هي باستخدام الـRPC(Remote Procedure Call) أي إستدعاء الاجراءات عن بعد بين الأغراض مثل الـDCOM والـCORBA، لكن الـHTTP غير مصمم لهذه الغاية. تشكل الـRPC مشكلة توافق بين التطبيقات إلى جانب المشكلة الأمنية حيث أن مخدمات الجدران النارية(firewalls) والوكلاء(proxy) تمنع عادةً مثل هذه الطلبات. إن أفضل الطرق للإتصال بين التطبيقات هي عبر الـHTTP وذلك لأنه متوافق مع جميع مستعرضات الوب والمخدمات. وتم بناء الـSOAP كمتممّ لذلك. ويؤمن الـSOP طريقة للإتصال بين التطبيقات التي تعمل على أنظمة تشغيل مختلفة، والمبنية بتقنيات ولغات برمجة مختلفة.

عناصر الـSOAP:

إن رسالة الـSOAP هي وثيقة XML عادية تحتوي العناصر التالية:

- ظرف الرسالة (Envelope):
  ويعرّف وثيقة الـXML كرسالة SOAP.
- رأس الرسالة(Header):
  وتحتوي على معلومات رأس الرسالة.
- محتوى الرسالة(Body):
  وتحتوي معلومات الإستدعاء والإستجابة.
- قسم الأخطاء(Fault element):
  وتحتوي معلومات عن الأخطاء والحالة.

يتم التصريح عن كل العناصر السابقة تحت فضاء عمل إفتراضي في ظرف الرسالة هو(http://www.w3.org/2001/12/soap-envelope) وفضاء العمل الافتراضي لأنماط المعطيات ولتشفير الـSOAP هو (http://www.w3.org/2001/12/soap-encoding).

قواعد جمل الـSOAP:

على رسائل الـSOAP:

- أن تتم كتابتها باستخدام الـXML؛
- أن تستخدم فضاء عمل ظرف الـSOAP؛
- أن تستخدم فضاء عمل تشفير الـSOAP؛
- أن لا تحتوي على تعليمات معالجة للـXML.

بنية رسالة الـSOAP:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
...
</soap:Header>
<soap:Body>
...
  <soap:Fault>  ...
  </soap:Fault>
</soap:Body>
</soap:Envelope>
```

1- ظرف الرسالة (SOAP Envelope Element):

يشكّل هذا العنصر جذر رسالة الـSOAP ويعرّف وثيقة الـXML كرسالة SOAP.

مثال:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  ...
  Message information goes here ...
</soap:Envelope>
```

2- رأس الرسالة(Header Element):

وهو عنصر إختياري من الممكن عدم إضافته إلى الرسالة، وفي حال وجوده فهو يحتوي على معلومات خاصة بالتطبيق مثل (التحقق من الصحة، دفع،..) ويجب أن يكون أول أولاد ظرف الرسالة في بنية وثيقة الـXML.

يجب أن يتم تمييز كل العناصر ضمن رأس الرسالة بفضاء عمل (namespace).

مثال:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
  <m:Trans xmlns:m="http://www.w3schools.com/transaction/"
  soap:mustUnderstand="1">234
  </m:Trans>
</soap:Header>
...
...
</soap:Envelope>
```

3- جسم الرسالة(Body Element):

يحتوي جسم الرسالة على الرسالة الفعلية المطلوب إرسالها إلى الطرف الآخر. وأيضاً يجب أن يتم تمييز كل العناصر ضمن جسم الرسالة بفضاء عمل (namespace).

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body>
  <m:GetPrice xmlns:m="http://www.w3schools.com/prices">
    <m:Item>Apples</m:Item>
  </m:GetPrice>
</soap:Body>
</soap:Envelope>
```

يمثل المثال السابق رسالة طلب سعر "التفاح"، ونلاحظ بأن m:GetPrice وm:Item هي عناصر خاصة بالتطبيق ولا تنتمي إلى فضاء عمل الـSOAP.

من الممكن أن تكون رسالة الإستجابة على الطلب السابق بالشكل التالي:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body>
  <m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">
    <m:Price>1.90</m:Price>
  </m:GetPriceResponse>
</soap:Body>
</soap:Envelope>
```

4- قسم الأخطاء(Fault Element):
وهو جزء إختياري يستخدم للإشارة إلى رسائل الخطأ إن وجدت. وفي حال وجود هذا الجزء فيجب أن يكون كإبن (child) ضمن جسم الرسالة.

لهذا الجزء أجزاء جزئية (sub elements) موضحة في الجدول التالي:

العنصر الجزئي	الوصف
<faultcode>	وهو ترميز لتعريف الخطأ
<faultstring>	شرح نصي مقروء
<faultactor>	معلومات عن مسبب الخطأ أو المشكلة
<detail>	ويحتوي على معلومات تفصيلية ومحدد خاصة بالتطبيق

ترميز الأخطاء:

يجب أن تكون القيم التي يمكن للعنصر الجزئي "faultcode" أن يأخذها من ضمن القائمة التالية:

الخطأ	الوصف
VersionMismatch	في حال العثور على فضاء عمل غير صحيح لظرف الـSOAP
MustUnderstand	للإشارة إلى عدم القدرة على فهم الرسالة
Client	تم تشكيل الرسالة بطريقة خاطئة أو أن الرسالة تحتوي على معلومات غير صحيحة
Server	يوجد مشكلة لدى المخدم ولا يمكن معالجة الرسالة

مثال عن الـSOAP:

في هذا المثال، يتم إرسال طلب الحصول على سعر الأسهم إلى المخدم "GetStockPrice". لهذا الطلب بارامتر هو إسم السهم "StockName" وبارامتر السعر "Price" الذي سيتم إرجاعه من قبل المخدم ضمن رسالة الاستجابة للطلب. فضاء عمل هذا التابع هو"http://www.example.org/stock":

رسالة الطلب:

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPrice>
    <m:StockName>IBM</m:StockName>
```

```
  </m:GetStockPrice>
</soap:Body>
</soap:Envelope>
```

رسالة الإستجابة:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPriceResponse>
    <m:Price>34.5</m:Price>
  </m:GetStockPriceResponse>
</soap:Body>
</soap:Envelope>
```

يمكن القول وبشكل مختصر بأن الـ SOAP=HTTP + XML

2-1-3. دليل الخدمات:

وهو اختصار لـUniversal Description،Discovery and Integration ، وهو عبارة عن إطار عمل(framework) مستقل عن منصة العمل(platform) لتوصيف الخدمات، واستكشاف الأعمال، ومكاملة خدمات الأعمال من خلال الانترنت، ويتم من خلاله تسجيل خدمات الوب والبحث عنها.

- إنه عبارة عن دليل يتم فيه تسجيل معلومات عن خدمات الوب؛
- إنه عبارة دليل واجهات خدمات الوب الموصفة باستخدام الـWSDL؛
- يتواصل باستخدام بروتوكول الـSOAP؛
- مبني على منصة عمل Microsoft .NET.

يستخدم الـUDDI مجموعة معايير من W3C وIETF مثل بروتوكولات XML،HTTP، and DNS .

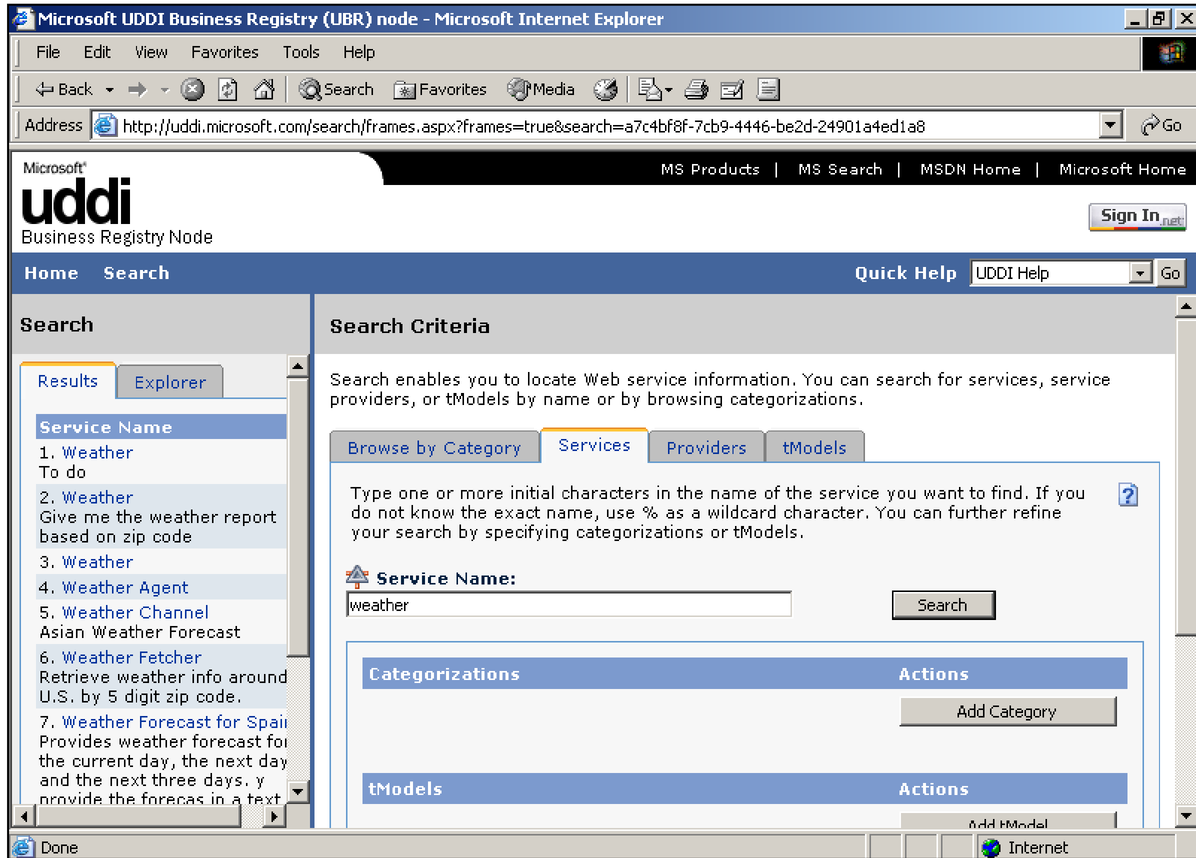
فوائد الـUDDI:

يمكن للأعمال والصناعات من مختلف الأحجام الإستفادة من الـUDDI. قبل وجود الـUDDI، لم يكن هناك معيار انترنت(Internet Standard) يسمح للأعمال بالوصول بمعلومات عن المنتجات والخدمات إلى الزبائن والشركاء. ويمكن للـUDDI أن يساهم في حل المشكلات التالية:

- يجعل من السهل إيجاد الأعمال الصحيحة التي يتم البحث عنها ضمن الملايين الموجودة على الانترنت؛
- الوصول إلى زبائن جدد وزيادة الوصول إلى الزبائن الحاليين؛
- يغني السوق بالعروض؛

- توصيف الخدمات وإجراءات العمل بطريقة برمجية في بيئة مفتوحة، وحيدة، وآمنة.

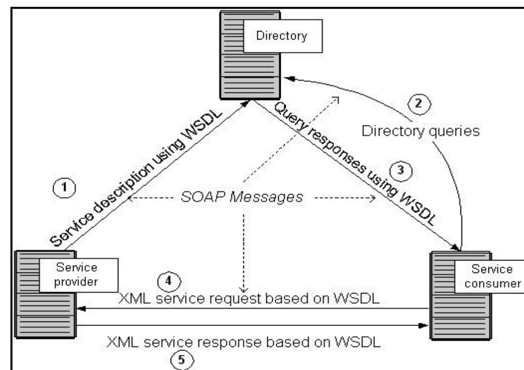
يوضح الشكل التالي "2-1" موقع دليل الخدمات على الإنترنت التي يتم من خلاله البحث عن الخدمات:



الشكل 2-1، موقع دليل الخدمات على الإنترنت

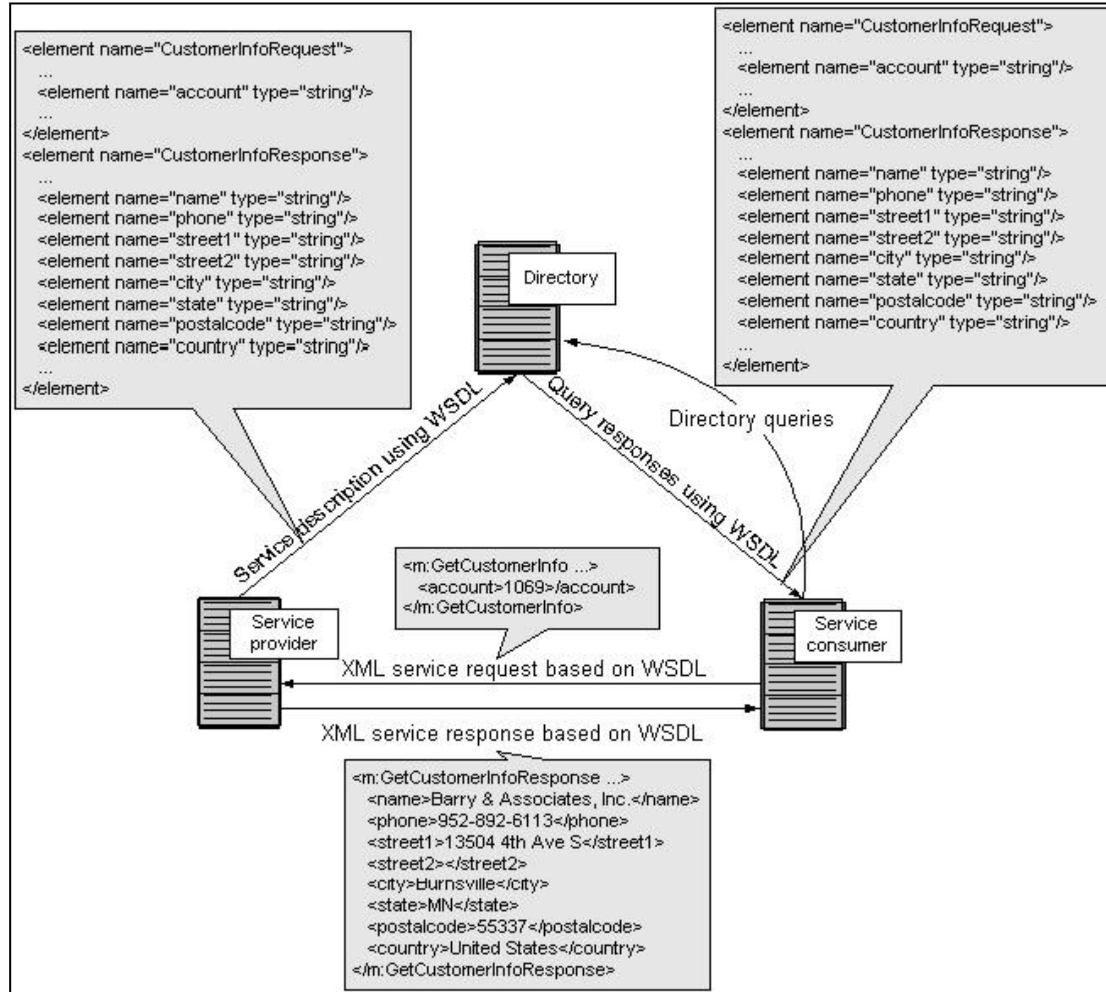
استخدام الـUDDI:

يمكن لعارض الخدمة أن يقوم بتسجيل خدماته في دليل الخدمات(UDDI) مثل حجوزات الطيران والفنادق. تقوم عندها وكالات السفر بالبحث ضمن دليل الخدمات عن الجهات التي تقدم خدمة الحجوزات. عندما يتم العثور على واجهة تعريف الخدمة(service interface) تقوم الوكالة بالتواصل مع الخدمة بشكل مباشر وذلك كونها تستخدم واجهة جيدة التعريف. الشكل التالي "2-2" يوضح هذه الخطوات:



الشكل 2-2، خطوات إستخدام خدمات الوب ودليل الخدمات

أما الشكل التالي "2-3" فيوضح واجهة توصيف الخدمة باستخدام الـWSDL:



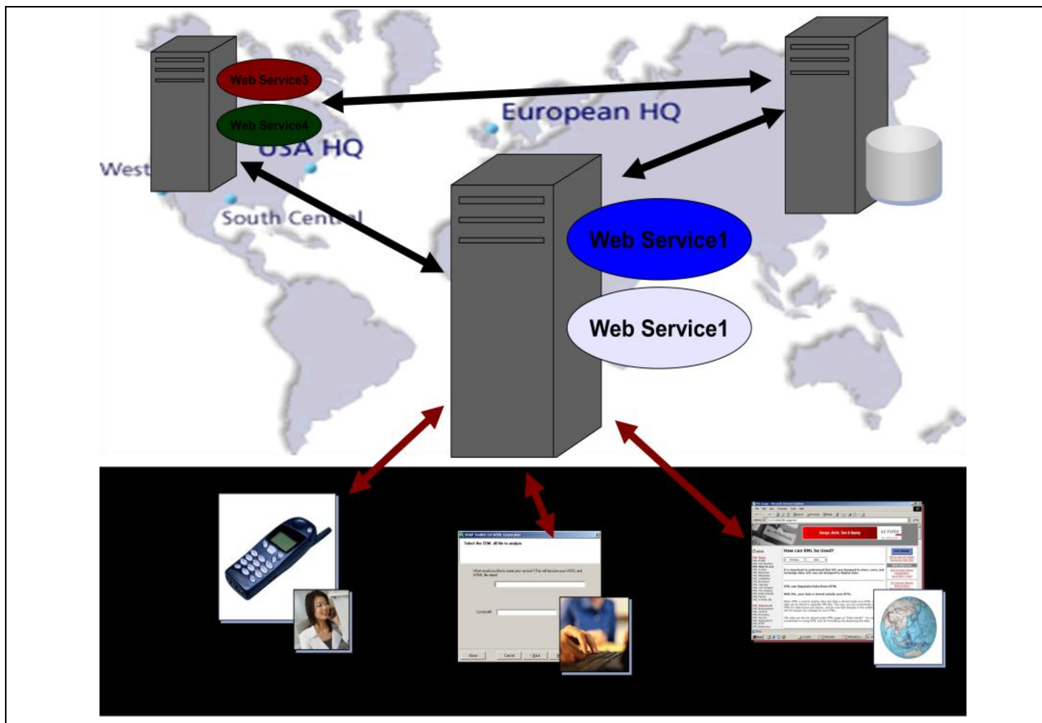
الشكل التالي 2-3، واجهة توصيف خدمة وب باستخدام الـWSDL

2-2. خدمات الوب والبنى خدمية التوجه:

يحقق إستخدام خدمات الوب في البنى خدمية التوجه مجموعة من الأهداف من [Barry 2003]:

1- تبني معايير قياسية: تتضمن هذه المعايير خدمات الوب وتعاريف العناصر المعيارية في الـXML.
2- إمكانية إستخدام أكبر للبرمجيات المعروضة والمتوافرة في السوق حيث تقوم البرمجيات بتزويد وصلات(adapters) لخدمات الوب التي تقدمها.
3- تغليف التطبيقات القديمة المستخدمة بواجهة تحقق المعايير وذلك باستخدام خدمات الوب كواجهة للتخاطب مع هذه التطبيقات والأنظمة السابقة.
4- إستخدام طبقة مستقلة عن البيانات بين التطبيقات والبيانات لإخفاء بنية البيانات، حيث تتم كل التعاملات كلها عن طريق خدمات الوب.

يوضح الشكل "2-4" مخطط ربط مجموعة من خدمات الوب في أماكن مختلفة حول العالم لتشكيل بنية خدمية التوجه يمكن استعراضها من خلال تطبيقات مكتبية أو تطبيقات وب أو تطبيقات على أية أجهزة طرفية يمكنها التعامل مع أنظمة المعلومات.



الشكل التالي 2-4، ربط مجموعة من خدمات الوب

2-2-1. مراحل تبني خدمات الوب والبنى خدمية التوجه:

ستمر معظم المنظمات بالمراحل التالية لتبني خدمات الوب والبنى خدمية التوجه:

1- إكتساب خبرة خدمات الوب:

وهي الفترة المطلوبة لاكتساب الخبرات في مجال خدمات الوب ولاسيما على مستوى تطوير خدمات الوب. يمكن لذلك أن يشمل محاولة اختبار إضافة خدمات وب خارجية إلى الموقع الإلكتروني للمنظمة، إمتلاك مخدم تطبيقات(Application Server) يستطيع الوصول إلى الأنظمة الموجودة باستخدام خدمات الوب، أو إختبار تعديل أنظمة لتبادل البيانات فيما بينها باستخدام خدمات الوب. كنتيجة لاختبار كل هذه الحالات سترتقي المنظمة إلى مرحلة أفضل من النضج اتجاه فهم مفاهيم خدمات الوب وأدوات تطويرها.

2- تحويل الأنظمة السابقة لاستخدام خدمات الوب:

هذا سيجعل من السهل تركيب خدمات الوب إلى الأنظمة السابقة. يملك العديد من مزودي الخدمات الأدوات اللازمة لتسهيل عملية التحويل هذه.

3- إلغاء حالات الإعتمادية بين الأنظمة:

إن هذه الإعتمادية ستضع القيود على مرونة البنى خدمية التوجه.

4- البدء بتطبيق البنية خدمية التوجه داخلياً:

ويتم ذلك من خلال تحديد التصميم الذي يرسم حدود كل خدمة من الخدمات في هذه البنية.

5- توسعة بنى خدمات الوب الداخلية لإضافة خدمات خارجية:

من الممكن أن يتم ذلك في أي وقت تظهر فيه أهمية إضافة هكذا خدمات خارجية كأن تتم إضافتها إلى الموقع الالكتروني للمنظمة.

نظرة إلى المستقبل:

إن خدمات الوب والبنى خدمية التوجه ستعني إنخفاض عدد الأشخاص العاملين بتقنيات المعلومات. حيث سيتركز عمل هؤلاء على خلق البنى إلى جانب خلق الوصلات اللازمة للعمل مع مجموعات الخدمات. وفي نفس الوقت ستتحسن جودة البرمجيات. كما أنه سيتم وضع معايير لإمكانيات الخدمات المتنوعة ولتشبيه ذلك بالمعايير القياسية الصناعية يمكن بشكل مشابه لمثال خدمة العرض رقم 2 الموضح بالشكل "1-5" الذي ذكرناه في الفصل الأول فإن خلق معايير للخدمات سيجعل من السهل شراء مجموعة من الخدمات وربطها مع بعضها البعض. مما يعني تناقص عدد العاملين على خلق البرمجيات أكثر وأكثر. إلا أن المنظمات ستحتاج للعاملين في مجال تقنيات المعلومات بهدف خلق الخدمات الخاصة التي يريدون تقديمها.

مع هذه النظرة المستقبلية لإيجاد معايير للخدمات سيكون من السهل إستبدال مجموعة من الخدمات بمجموعة خدمات أخرى (إن هذا سيشبه إستبدال المُستقبِل في المثال المذكور بمُستقبِل آخر يملك مواصفات أفضل). مما سيدفع مزودي الخدمات إلى المنافسة لتحسين جودة منتجاتهم. حيث لن يكون هناك مانع للمنظمات من استبدال المنتجات التي يستخدمونها بأخرى أكثر جودةً بسبب سهولة الإستبدال.

2-2-2. تكلفة التحول إلى البنى خدمية التوجه وخدمات الوب:

إن التحول إلى البنية خدمية التوجه سيعني إجراء مجموعة من التغييرات داخل المنظمات عبر مراحل محددة، سيكون لهذه المراحل تكلفة:

1- اكتساب خبرة خدمات الوب:

يتم ذلك من خلال إشراك مجموعة من الأشخاص في مشاريع صغيرة للإطلاع بشكل أكبر على تطوير خدمات الوب. سيتطلب ذلك تكلفة التدريب وشراء بعض الأدوات البرمجية بالإضافة إلى الوقت اللازم للتدريب والتجريب.

2- تحويل الأنظمة السابقة لاستخدام خدمات الوب:

يعتمد ذلك على البنية الموجودة حيث ستتطلب هذه المرحلة زيادة العمل من قبل العاملين في تقنيات المعلومات لدى المنظمة وسيتطلب ذلك وقت وتدريب أو إستقدام كادر خارجي إضافي للقيام بهذه المهمة.

3- إلغاء حالات الاعتمادية بين الأنظمة:

في حال وجود مجموعة من الأنظمة الداخلية التي يجب أن تتحول إلى خدمات منفصلة ولكنها تتشارك بنى المعطيات أو تتشارك موارد أخرى، فستكون هناك حاجة للاستثمار في إلغاء هذه

الاعتمادية فيما بين النظم. يمكن أن يضاف لذلك تكلفة التحليل اللازمة لتحديد العلاقات التي تتطلب الإلغاء بين النظم. بالطبع ليس من الضروري إلغاء العلاقات التي لا تشكل عائقاً أمام مرونة البنى خدمية التوجه.

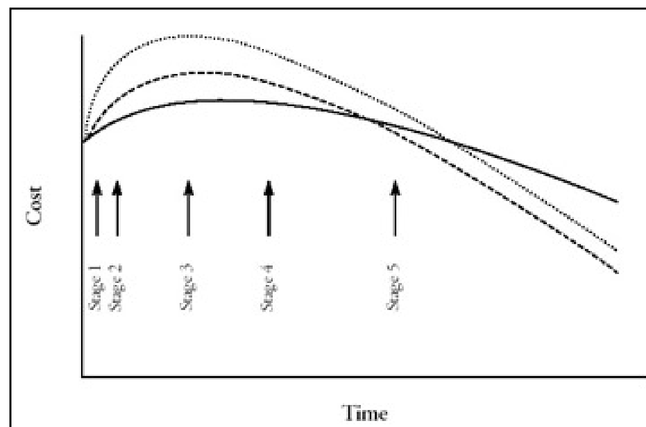
4- البدء بتطبيق البنية خدمية التوجه داخلياً:

بعد إلغاء اعتمادية الأنظمة الداخلية على بعضها البعض، يمكن البدء ببناء النظم خدمية التوجه وهنا ستبدأ التكلفة بالانخفاض وذلك بسبب انخفاض تكلفة الصيانة بشكل رئيسي بسبب استخدام خدمات الوب كتقنية عامة للاتصال.

5- توسعة بنى خدمات الوب الداخلية لإضافة خدمات خارجية:

بعد تطبيق البنية خدمية التوجه يمكن البدء ببناء برمجيات بالاعتماد على تركيب مجموعة من الخدمات. يمكن ملاحظة انخفاض التكلفة في هذه المرحلة لانخفاض تكلفة التطوير.

يُظهر المخطط التالي "5-2" نظرة إلى التكلفة مقارنة بالزمن اللازم لتبني المنظمة للبنية خدمية التوجه. بالطبع إن هذا سيعتمد على طبيعة بنية الأنظمة الداخلية السابقة في الشركة.



الشكل 2-5، تكلفة تبني خدمات الوب مع الزمن

3-2-2. المزيد عن خدمات الوب والبنى خدمية التوجه:

بالعودة إلى النظم خدمية التوجه التي قمنا بتعريفها والحديث عن خصائصها ومواصفاتها بشكل مفصل سابقاً من خلال الفصل الأول، وفي زيارة مجددة لمثال خدمة الشراء الإلكتروني رقم 4 الذي يقوم بإستخدام مجموعة من خدمات الوب كما في الشكل "1-7"، حيث يبدو من الواضح بأن تشكيل هذه البنية يتم بطريقة يدوية وبمجهود بشري. قد لا يشكل هذا مشكلة بالنسبة لمثال من هذا النوع لأنه غالباً سيتم إستخدام نفس الخدمات بعد تجميعها وإعدادها من أول مرة، لكن بالنسبة لمستهلكي خدمة أخرين قد يكون من الضروري تحديد الخدمات ومعرفة طريقة التخاطب معها بشكل أكثر ديناميكية وبحاجة أقل للتدخل البشري، أي أن نفس الطلب قد يرسل إلى النظام فيقوم النظام في كل مرة بإستخدام مزود خدمة مختلف لتلبية هذا الطلب.

كما ويمكن أن يؤثر توافر الخدمة المستهلكة (Service Availability) على الوظيفة الكلية للنظام خدمي التوجه، لذلك سيكون من الضروري أن يتمكن النظام من البحث بشكل أتوماتيكي على خدمة بديلة ليقوم بتسجيلها كمزود خدمة وإستخدامها لضمان استمرارية العمل. قد يكون من الصعب أحياناً

إيجاد الخدمة المطلوبة لبناء النظام خدمي التوجه بشكل مباشر، في الوقت الذي يكون من الممكن إيجاد تركيبة من تفاعل مجموعة من الخدمات لتوفير الخدمة النهائية المرجوة، وهي ما يمكن أن نسميه بالخدمة الإفتراضية (Virtual Service).

إن من المقاربات الموجودة لحل جزء من هذه المشاكل هو استخدام الوكلاء البرمجيين (Software Agents)، حيث يعرف الوكيل البرمجي بأنه "برنامج يعمل لتحقيق هدف معين في بيئة متغيرة بدون تدخل مستمر للعامل البشري وبأقل مقدار من التحكم لتنفيذ المهام الموكلة إليه" [Tolle 2001]. وتعتمد هذه المقاربات على بناء وكلاء برمجيين لدى كل من مزودي ومستهلكي الخدمة، ويتم تعريف طريقة الإتصال والتفاوض بهدف الوصول إلى الغاية النهائية.

لعل أهم التحديات التي قد تواجهنا في حل مشاكل من هذا النوع هوأن تكون الخدمات المتوافرة مفهومة بالنسبة للآلة، وذلك حتى يتمكن النظام من إيجادها، ويحدد طريقة التخاطب معها، وتضمينها في بنيته، وبالتالي تقديم الخدمة النهائية المرجوة، ومن هنا ندرك أهمية وجود الوب الدلالي كتطور مهم في عالم تقنيات الوب كمفهوم يساعد على التغلب على هذا النوع من المشاكل. وهذا ما سيتم التطرق إليه بتفصل أكبر من خلال الأجزاء التالية.

3-2. خلاصة:

لقد بحثنا في هذا الفصل ظهور خدمات الوب كتطور مهم في تقنيات الوب والذي ينتقل بالوب من الجانب الستاتيكي إلى الجانب الديناميكي، بالإضافة إلى أهمية إستخدام خدمات الوب في النظم خدمية التوجه. ولقد هدفنا من خلال هذا الفصل إلى الوقوف على الطريقة الأكثر إستخداماً لتحقيق الأنظمة خدمية التوجه وما يحله ذلك من مشاكل. قمنا من خلال هذا البحث بتحديد التحديات التي تواجه هذه المقاربة واستنتاج ما تعانيه من نقص.

الجزء الثاني

الوب الدلالي

تمثيل المعرفة والأنطولوجية في الوب الدلالي

Knowledge Representation & Ontology

In Semantic Web

مقدّمة:

يهتم تمثيل المعرفة بدراسة تمثيل الحقائق والمعارف بشكل صوريّ ومعالجتها من قبل الآلة. حيث تسمح تقنيات الإستدلال الآلي في أنظمة الكمبيوتر بالوصول إلى استنتاجات جديدة بالإعتماد على المعرفة المُمثّلة بشكل قابل للفهم من قبل الآلة. ظهرت الأنطولوجية مؤخراً في علوم الحاسب والذكاء الصنعي لتزويد أنظمة الحاسب بنماذج مفاهيمية تتعلق بما هو موجود في مجالٍ أو ميدانٍ ما. بهذه الطريقة يمكن لأنظمة الكمبيوتر اتخاذ قرارٍ ما باستخدام الإستدلال عن المجال الذي تمثله المعرفة بشكل يحاكي طريقة التفكير والإستنتاج البشري [Sowa 2000]. سنتطرق في هذا الفصل إلى المفاهيم الأساسية في تمثيل المعرفة والأنطولوجية التي يجري استخدامها في أنظمة الحاسب، إلى جانب لغات الأنطولوجية المُستخدمة في سياق الوب الدلالي.

3-1. تمثيل المعرفة:

إن تمثيل المعرفة والإستدلال يهدفان إلى تصميم أنظمة كمبيوتر قادرة على الإستنتاج والإستنباط بشكل يحاكي طريقة التفكير البشري من خلال إيجاد طريقة لتمثيل الحقائق والمعارف من خلال مجموعة من الرموز. حيث تحتفظ الأنظمة المعتمدة على المعرفة بقاعدة بيانات تسمى قاعدة المعرفة (Knowledge Base) وتشمل المعارف المتعلقة بمجال ما، وتقوم الأنظمة باتخاذ القرار من خلال طرح أسئلة متعلقة بالمجال الذي يجري تمثيله على قاعدة المعرفة هذه [Sowa 2000].

3-1-1. طرق تمثيل المعرفة:

يتم تمثيل المعرفة بعدة طرق ومنها الشبكات الدلالية (Semantic Networks)، والقواعد (Rules)، والمنطق (Logic). حيث يتم تمثيل الشبكات الدلالية من خلال مخططات الـRDF "Resource Description Framework"، بينما يتم تمثيل المعارف المتعلقة بالأعمال من خلال القواعد من الشكل "If Then" أو من خلال المنطق.

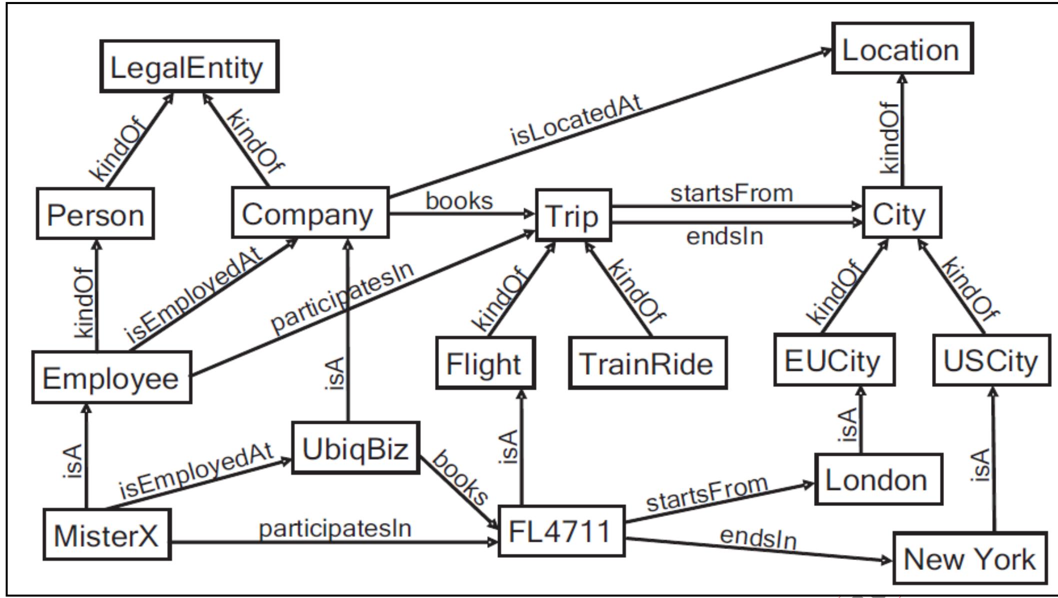
3-1-1-1. الشبكات الدلالية:

تعود نشأة الشبكات الدلالية إلى المخططات التي قدمها Charles Peirce في عام 1896 للتعبير عن مجموعة من التعابير المنطقية من خلال مخططات تتألف من مجموعة من العُقد. ثم تطورت لاحقاً باستخدام رموز ذات دلالات مختلفة تُجمِع على تصنيف الأغراض وتوصيف العلاقات فيما بينها.

الشبكة الدلالية وهي عبارة عن مخطط بياني يتألف من مجموعة من العُقد والأسهم حيث تمثل العُقد المفاهيم (Concepts) المراد تمثيلها والأسهم تمثل العلاقات (Relations) بين هذه المفاهيم. بحيث يمثل هذا المخطط مجموعة التعابير التي تمثل مجال ما. فعلى سبيل المثال يمكننا في تمثيل المجال المتعلق برحلات العمل الذي يهتم بالتخطيط لرحلات العمل لمجموعة من الموظفين إلى وجهة ما، يمكننا إيجاد مجموعة من المفاهيم مثل المفهوم "موظف"، "شركة"، و"طيران"، كما يمكن إيجاد مجموعة من العلاقات مثل "حجز"، "موظف عند"، و"مشترك بـ".

تقوم الشبكات الدلالية بتمثيل المعرفة المشتقّة من اللغة الطبيعية بشكل قابل للفهم من قبل الحاسبات. إن المعارف الممثّلة بالشبكة الدلالية في المخطط "3-1" تتطابق مع محتوى النص التالي المكتوب باللغة الطبيعية:

"الموظفين في الشركات هم أشخاص، قد تقوم الشركة بحجز رحلة عمل لهم، يمكن أن تكون الرحلة بواسطة القطار أوالطائرة والتي تبدأ وتنتهي في مدن أوروبية أو أمريكية. إن شركة UbiqBiz تحجز في رحلة الطيران FL4711 والتي تنطلق من مدينة لندن إلى نيويورك للموظفMisterX".



الشكل 3-1 مثال عن الشبكة الدلالية

عادةً ما يتم اختيار الأسماء الموجودة في النص لتعبّر عن المفاهيم والأفعال تمثّل العلاقات بين هذه المفاهيم. فمثلاً تتم قراءة $\boxed{\text{Company}} \xrightarrow{books} \boxed{\text{Trip}}$ بالشكل "Companies book trips" كعلاقة ثنائية بين مفهومين.

تُظهر الشبكة الدلالية السابقة العلاقة بين المفهوم العام "Employee" والمفهوم الخاص "MisterX" حيث يعبّر المفهوم الأول عن فئة من مجموعة من الأغراض التي تمتلك نفس المواصفات وهي مجموعة كل الموظفين، أما المفهوم الثاني فيعبّر عن عنصر أو غرض محدد ضمن المجال الذي يتم توصيفه. إن العلاقة التي تربط بين العنصر والفئة التي ينتمي إليها هي العلاقة $\xrightarrow{isA}$ ويُظهر الجزء السفلي من الشبكة الدلالية السابقة المعارف الخاصة بمجموعة من العناصر التي تعكس الوضع الحالي للعنصر "MisterX" الذي يشترك في رحلة طيران معينة، أما الجزء العلوي فيظهر مجموعة من المفاهيم العامة التي تعكس حالات أخرى ممكنة.

من الممكن ملاحظة علاقات أخرى ضمن الشبكات الدلالية ومنها علاقة التخصيص "Subsumption" والتي يشار إليها بـ $\xrightarrow{kindOf}$ ، حيث تربط هذه العلاقة بين مفهومين عامين للدلالة على علاقة التخصيص أو التعميم فيما بينهما ففي الشبكة السابقة يظهر المفهوم "Flight" كنوع خاص من المفهوم "Trip" مما يعني بأن الطيران هو رحلة أيضاً، على الرغم من وجود رحلات أخرى غير الطيران كالقطار مثلاً. يتم فهم هذه العلاقة كعلاقة الوراثة التي يرث فيها المفهوم الخاص كل المواصفات التي يتصف بها المفهوم الأب ففي الشبكة السابقة يمكننا القول بأن شركة ما يمكن أن تقع في مدينة أوروبية ما حيث تشير العلاقة "LocatedAt" إلى العلاقة من المفهوم "Company" إلى المفهوم "Location" وحيث أن المفهوم "EUCity" هو نوع من المفهوم "City" والذي يمثل أحد أنواع المفهوم "Location". ومن العلاقات الأخرى في الشبكات الدلالية $\xrightarrow{partOf}$ والتي تشير إلى علاقة جزء من كل.

يبقى أن نشير إلى أن الشبكات الدلالية تشبه نوع آخر من أنواع تمثيل المعرفة المعروف بنظام الصناديق (Frame Systems) والذي يتم فيه تمثيل المفاهيم ضمن صناديق تحتوي على منافذ تمثل العلاقات حيث ترتبط الصناديق مع بعضها بملئ هذه المنافذ للتعبير عن ارتباط المفاهيم ببعضها البعض.

إن الشبكات الدلالية تعتبر خياراً جيداً لتمثيل المعارف في المجالات التي يتم يمكن فيها تصنيف المفاهيم ضمن فئات في الوقت الذي لا تدعم الشبكات الدلالية تمثيل أنواع المعطيات كالنمط النصي والرقمي وغيرهما من الأنماط.

3-1-1-2. القواعد:

وتُعتبر من الطرق التي يتم من خلالها تمثيل المعارف الخاصة بمجال ما. وتأتي بالشكل "If-Then" مما يسمح بالتعبير عن عبارات أكثر تعقيداً. كما ويمكن إيجاد هذه القواعد في أنظمة البرمجة المنطقية مثل Prolog، قواعد البيانات الاستباطية، أو أنظمة قواعد العمل. يوضح المثال التالي كيفية التعبير عن المعرفة الخاصة برحلات العمل من خلال مجموعة من القواعد:

(1) IF *something is a flight* THEN *it is also a trip*
(2) IF *some person participates in a trip booked by some company*
THEN *this person is an employee of this company*
(3) FACT *the person MisterX participates in a flight booked by the company UbiqBiz*
(4) IF *a trip's source and destination cities are close to each other*
THEN *the trip is by train*

تتألف كل قاعدة من جسم (Body) والذي يمثل الشرط، ورأس (Head) وهو الجزء الذي يأتي بعد العبارة "Then". إن النظم المعتمدة على تمثيل المعرفة باستخدام القواعد عادةً ما تعمل على الحقائق (Facts) والتي يتم التعبير عنها بدون استخدام جسم القاعدة "أي أن النتيجة واقعة بدون وجود أي شرط مما يجعلها حقيقة"، حيث تبدأ هذه الأنظمة من خلال مجموعة من الحقائق ثم تطبق القواعد المعطاة لاستنباط حقائق جديدة.

إن كتابة القواعد بالطريقة السابقة لن تكون مفهومة بالنسبة للآلة لذلك يتم كتابتها بأحد لغات القواعد بالشكل التالي:

```
(1) Trip(?t) :- Flight(?t)
(2) Employee(?p) ∧ isEmployedAt(?p,?c) :-
    Trip(?t) ∧ books(?c,?t) ∧ Company(?c)∧
    participatesIn(?p,?t) ∧ Person(?p)
(3) Person(MisterX) ∧ participatesIn(MisterX,FL4711)∧
    Flight(FL4711) ∧ books(UbiqBiz,FL4711) ∧ Company(UbiqBiz) :-
(4) TrainRide(?t) :-
    Trip(?t) ∧ startsFrom(?t,?s) ∧ endsIn(?t,?d) ∧ close(?s,?d)
```

تتم في معظم أنظمة البرمجة المنطقية قراءة القاعدة بالعكس "من اليمين إلى اليسار". ويتم إسناد القيم إلى المتغيرات التي تظهر أمامها العلامة "?". حيث إن القاعدة الأولى تُظهِر علاقة الوراثة بين الرحلة ورحلة الطيران حيث تعتبر كل رحلة طيران هي رحلة. أما القاعدة الثانية فتوضِّح كيفية إستنتاج الإشتراك برحلة طيران. يمكن بالإعتماد على الحقائق المذكورة في القاعدة الثالثة إستنتاج بأن MisterX هو موظف في الشركة UbiqBiz. أما القاعدة الرابعة فيمكن تفسيرها كجزء من سياسة السفر الخاصة ببعض الشركات والتي تقول بأن السفر يجب أن يكون باستخدام القطار إذا كانت المدينتين قريبتين.

3-1-1-3. المنطق:

في الطريقتين السابقتين تم استخدام المنطق في التمثيل لإعطاء دلالة دقيقة لما يتم تمثيله وبدون ذلك سيكون التمثيل غامضاً. إن مخطط الشبكة الدلالية الذي تم عرضه سابقاً لا يمكننا من معرفة فيما إذا كانت الشركة تستطيع إجراء الحجوزات لموظفيها فقط أم لموظفيها وموظفي الشركات الشريكة لها. ولا يمكننا معرفة فيما إذا كان بإمكان أية شركة القيام بالحجز أم بعض الشركات فقط. أيضاً فإن الطريقة الثانية "القواعد" وبالرغم من أنها أكثر ظهوراً، فإن المعنى الحقيقي يبقى غير واضح في حالات النفي التي قد تستخدم في بعض القواعد والتي قد تقود إلى نتائج متضاربة.

إن أكثر طرق التمثيل المنطقي هي الـ"first-order predicate calculus" أو " first-order logic". وسنعرض من خلال الأمثلة التالية كيف يتم تمثيل الشبكة الدلالية التي استعرضناها سابقاً من خلال الـfirst-order logic. فعلاقة الوراثة يمكن كتابتها بالشكل التالي:

Employee —kindOf→ Person	$\forall x : (Employee(x) \rightarrow Person(x))$

حيث يعبّر المتغير x عن كل الأغراض الموجودة في المجال الموصّف، وتتم قراءة العبارة بالشكل التالي "كل ما هو موظف هو شخص". سنوضّح في المثال التالي كيف يمكن وضع قيود إضافية باستخدام التمثيل المنطقي:

Company —books→ Trip	$\forall x, y : (books(x,y) \rightarrow Company(x) \wedge Trip(y))$ $\forall x : \exists y : (Trip(x) \rightarrow Company(y) \wedge books(y,x))$

في الوقت الذي تغيب فيه بعض التفاصيل في طريقة التمثيل باستخدام المخططات البيانية كما في الشبكات الدلالية، فإنه يمكن تحديد العلاقة بين المفاهيم بشكل أكثر دقة فالعبارة الأولى السابقة تقول بوجود علاقة "الحجز" بين أي رحلة وأي شركة، بينما العبارة الثانية تؤكد بأن لكل رحلة تم حجزها يوجد شركة قامت بذلك الحجز.

إذاً من الممكن تحديد قيود أكثر تعقيداً تعطي تمثيلاً أكثر دقةً من خلال التمثيل المنطقي فمثلاً إن التمثيل التالي للعلاقة بين المفاهيم "الموظف"، "الشركة"، و"الرحلة" يقول بأنه من أجل كل رحلة يجب أن يكون هناك موظفاً مشتركاً فيها بحيث أن الشركة التي يعمل لديها هذا الموظف هي من قام بإجراء عملية الحجز هذه.

Company —books→ Trip ←participatesIn— Employee ; Employee —employedAt→ Company

$$\forall x : \exists y : (Trip(x) \rightarrow Employee(y) \wedge participatesIn(y,x) \wedge books(employer(y),x))$$

إن إمكانية التمثيل المنطقي لا تقتصر فقط على ما يمكن تمثيله باستخدام الشبكات الدلالية وإنما يمكن توصيف القواعد بصيغ منطقية كما في المثال التالي:

IF *a trip's source and destination cities are close to each other*
THEN *the trip is by train*
is the translation to the logical formula

$$\forall x, y, z : (Trip(x) \wedge startsFrom(x, y) \wedge endsIn(x, z) \wedge close(y, z) \rightarrow TrainRide(x)).$$

يوجد عدة أنواع للتمثيل المنطقي للمعارف ومنها:

1- Description Logics
2- Closed-World Assumption
3- Non-monotonicity
4- Logic Programming
5- Higher-Order Logic

3-2. الأنطولوجية:

يعود مفهوم الأنطولوجية إلى الفلسفة، وهو من فروع ما وراء الطبيعة والذي يشير إلى الاستكشاف الفلسفي للوجود. ويهتم بسؤال أساسي وهو "ما الموجود؟" و"ما طبيعة الأشياء الموجودة؟" [Craig 1998]. بالعودة إلى أرسطو، فإن السؤال "ما الموجود؟" يقود إلى دراسة الفئات العامة (Categories) لكل الموجودات. إن هذه الفئات تعطي معنىً لتصنيف الأشياء، ويسمح تنظيم هذه الفئات بتحليل العالم المركب من هذه الأشياء بطريقة مهيكلة. تعرف هذه الفئات في الأنطولوجية بإسم "الكليّات" (Universals) أيضاً، وأما الأشياء فتعرف بالبنود (Particulars).

لقد اهتم الفلاسفة بالمستوى الأعلى من الفئات والتي تشمل العالم الفيزيائي ككل. ومن هذه الفئات: المادة، الأغراض الفيزيائية، الأغراض الغير ملموسة، وغيرها. بالعودة إلى تمثيل المعرفة وعلوم الحاسب، يمكن لنظم المعلومات الإستفادة من مفهوم التصنيف الأنطولوجية هذا بتطبيقة على مجال أو ميدان محدد في هذا العالم. وكمثال عن هذا التصنيف في ميدان رحلات العمل نجد الفئات التالية: "شخص"، "شركة"، "رحلة"، "رحلة طائرة"، أما بالنسبة للبنود فيمكن أن نجد "السيد س" من الفئة "شخص" و"الشركة س" من الفئة "شركة" و"الرحلة 110" من الفئة "رحلة طيران".

بشكل عام، إن عملية إختيار فئات الأنطولوجية والأغراض التي ضمنها في ميدان ما يحدد ما يمكن تمثيله في أنظمة الحاسب [Sowa 2000]. وهكذا فإن الأنطولوجية ستزودنا بأسماء العقد والأقواس في الشبكات الدلالية (Semantic Networks) أو بأسماء الإسنادات والثوابت في القواعد أو التعابير المنطقية. وبتحديد ما هو موجود يمكن معرفة ما يمكن استنتاجه. إن الأنطولوجية في علوم الحاسبات تهتم بترميز المعارف الخاصة بمجال عمل ما أو ميدان محدد بطريقة قابلة للفهم من قبل الآلة لجعلها متاحة أمام البرمجيات ونظم المعلومات.

3-2-1. تعريف الأنطولوجية:

يتم تعريف الأنطولوجية من وجهات نظر مختلفة وفقًا لمجال العمل أو الميدان الذي يجري فيه استخدام الأنطولوجية، ففي مجال الوب الدلالي يتم تعريف الأنطولوجية بأنها تصوير واضح لخصائص ومواصفات المفاهيم الموجودة في مجال أو ميدان ما [Gruber 1993]. يُضيء هذا التعريف على

عدة مواصفات للأنطولوجية كتوصيف المعارف الخاصة بمجال العمل، تمثيل المعارف بشكل صوري، التوصيف الصريح، قابلية للمشاركة، خاص بمفاهيم مجال ما.

- **التمثيل الصوري (Formality):**

يتم تمثيل الأنطولوجية باستخدام لغات تمثيل المعارف التي تعطي صورة دلالية. إن هذا يضمن لتوصيف المعارف الخاصة بالمجال الخاص بهذه الأنطولوجية أن يكون قابل للفهم من قبل الآلة ويتم تفسيره بطريقة جيدة التعريف.

- **التصريح (Explicitness):**

تقوم الأنطولوجية بتحديد المعارف الخاصة بالمجال بشكل واضح حتى تتمكن الآلة من التعامل معه. أية تعابير لا يتم تحديدها بوضوح لن تكون قابله للفهم من قبل الآلة وإن كانت من البديهيات بالنسبة للتفكير البشري.

- **قابلية للمشاركة (Being Shared):**

تعكس الأنطولوجية اتفاق الأشخاص في مجال ما على المفاهيم الخاصة بهذا المجال. بالتالي كلما كبر مجتمع هؤلاء الأشخاص أصبحت عملية الاتفاق والمشاركة بينهم أكثر صعوبة. ولهذا فإن الأنطولوجية تكون خاصة بمجموعة محددة ضمن مجتمع ما ويعتمد بناؤها على الاجماع.

- **المفاهيمية (Conceptuality):**

تقوم الأنطولوجية بتحديد المعارف عن طريق المفاهيم حيث يتم استخدام مجموعة من الرموز للتعبير عن المفاهيم والعلاقات فيما بينها. وقد تكون هذه المفاهيم والعلاقات من البديهيات لدى الانسان لأنها مرتبطة بالنموذج العقلي للتفكير البشري [Guarino 1998].

- **خاص بمجال عمل (Domain Specificity):**

إن التوصيف الذي تغطيه الأنطولوجية يكون خاصاً بمجموعة المعارف المتعلقة بمجال عمل ما. كلما كان مجال العمل أكثر محدودية كلما استطاع مهندس الأنطولوجية التركيز على التفاصيل أكثر من تركيزه على تغطية فضاءٍ أكبر من المواضيع ذات الصلة. يمكن بهذه الطريقة إجراء التوصيف لمجموعة المعارف على عدة أقسام يهتم كلٌ منها بالأنطولوجية الخاصة بمجال أكثر محدودية.

من الناحية التقنية تحتوي الأنطولوجية بشكل أساسي على مفاهيم، علاقات، وأمثلة عن هذه المفاهيم. ويمكن النظر إلى الأنطولوجية على أنها مجموعة من العبارات والتي يمكن أن تكون من البديهيات من مثل "فلان موظف" والتي تعبر عن مفهوم، ومثال عن هذا المفهوم. وقد تكون أكثر تعقيداً من مثل "يمكن لموظفي الشركة فقط التواجد في الرحلة التي حجزتها الشركة " والتي تظهر العلاقة بين مفهومين.

قد يبدو لنا أن نمذجة المفاهيم في الأنطولوجية مشابهة للنمذجة غرضية التوجه في هندسة البرمجيات أو مشابهة لتصميم مخططات قواعد البيانات العلائقية. في الحقيقة يوجد عدة اختلافات فيما بينها ومنها:

1- عادةً ما تؤمّن لغات الأنطولوجية توصيف دلالي أغنى من غرضية التوجه أو قواعد البيانات العلائقية حيث يتم ترميز المعلومات المعقّدة بناءً على سياقها وارتباطها المنطقي.

2- يتم تطوير الأنطولوجية لأغراض مختلفة ففي الوقت الذي يتم تطوير مخططات قواعد البيانات لحفظ البيانات ويتم تطوير نموذج الأغراض لتنفيذ العمليات من قبل الآلة، فإن الأنطولوجية تحدد مجموعة المعارف الخاصة بمجال عمل ما بشكل يسمح بالاستنباط والاستنتاج.

يمكن تلخيص ما سبق بأن الأنطولوجية المستخدمة في نظم المعلومات هي نموذج مفاهيمي لمجال عمل النظام. يكون هذا النموذج قابل للفهم والمعالجة من قبل الآلة باستخدام تقنيات تمثيل المعرفة وبالتالي يمكن للنظام اتخاذ قرارات بالاعتماد على الاستنتاج من هذه المعارف.

3-2-2. المكونات الأولية للأنطولوجية:

في [Gruber 1993] تم تعريف الأنطولوجية باستخدام الخماسية (C،I،R،F،A) حيث:

- C وهي مجموعة المفاهيم (Concepts) وتدعى أيضاً الصفوف أو الأصناف: أي هي مجموعة الأسماء المجردة (Abstraction) المستخدمة لتوصيف الأغراض في العالم؛
- I وهي مجموعة الأفراد (Individuals) أو الأغراض الفعلية في العالم. تدعى الأغراض أيضاً بأمثلة (Instances) المفاهيم؛
- R وهي مجموعة العلاقات المعرّفة على المجموعة C بحيث يكون كل علاقة R ∋ r هي حدودية مرتبة من الشكل (c1 X c2 X… X Cn)=r، مثلاً العلاقة subconcept-of هي عبارة عن زوج من المفاهيم (Cp،Cc) حيث يمثل المفهوم Cp المفهوم الأب والمفهوم Cc هي المفهوم الابن؛
- F وهي مجموعة التوابع (Functions) المعرّفة على المفاهيم والتي تعيد مفهوماً بالنتيجة، أي إن كل عنصر f من هذه المجموعة يكتب بالشكل: (C1 X C2 X … X Cn-1 → Cn) مثلاً التابع Price-Of-Flat يطبق على المفاهيم Square، Location، Year ويعيد المفهوم Price كنتيجة، أي:

  Price-Of-Flat: Year X Location X Square → Price
- A وهي مجموعة المسلمات (Axioms) التي تحدد المعنى المرجو من المفاهيم والعلاقات والتوابع.

ليس من الضروري تواجد جميع هذه المكونات في الأنطولوجية بل يكفي تواجد بعضها فقط، فمثلاً من النادر وجود الأمثلة (Instances) في معظم الأنطولوجيات. أبسط أنواع الأنطولوجيات (الأنطولوجيات الخفيفة) مكونة من مجموعة المفاهيم C ومجموعة العلاقات R.

يتم تنظيم المفاهيم والأمثلة بهرمية بحيث يكون نمط العلاقة فيما بينهم هو علاقة (is-a) والتي تسمح باستغلال مفهوم الوراثة بين المفاهيم أو الأمثلة. أي إذا كان A سلفاً لـB وكان B سلفاً لـC فإن A سلفٌ لـC. إن العلاقة (is-a) ليست الوحيدة التي يمكن تعريفها بين المفاهيم بل هناك علاقات أخرى يمكن تعريفها (مثل علاقة Parts-Of).

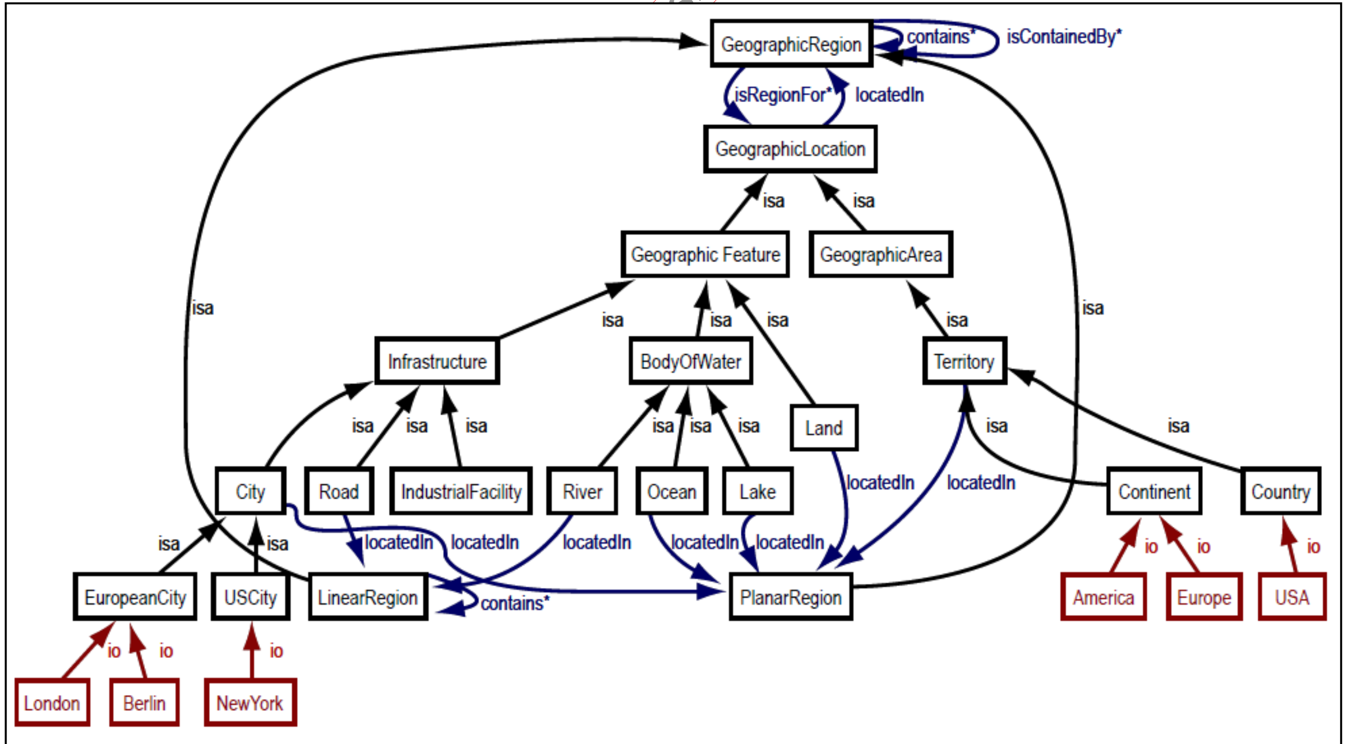
يتم تعريف المفاهيم من خلال الواصفات (Attributes or Slots) المميزة لها، وفي حال كانت المفاهيم منظمة بهرمية (is-a) فإن الوراثة ستمتد لتشمل واصفات المفاهيم المشكلة للهرمية، وبالتالي يضم كل مفهوم واصفاته الخاصة إضافةً إلى الواصفات الموروثة من المفهوم الأب.

3-2-3. ظهور الأنطولوجيات:

يقوم مهندس المعرفة بالتعامل مع الأنطولوجية كمجموعة من مخططات البيانات التي يتم ترميزها وتخزينها بالإعتماد على أحد لغات الأنطولوجية بشكل قابل للفهم من قبل الآلة. يقوم محرك الاستدلال (Reasoner) بدوره بتفسير الأنطولوجية كمجموعة من البديهيات التي تشكل الوصف المنطقي.

في سيناريو نظام رحلات العمل، سنلاحظ الحاجة إلى توصيف أكثر من مجال، فلكي يتمكن محرك الاستدلال من استنتاج معلومات للتخطيط للرحلة فهو بحاجة إلى مجموعة من المعارف عن البنية التحتية للقطارات، رحلات الطيران واستئجار السيارات من جهة، ومن جهة أخرى فهو بحاجة للتعامل مع مجموعة من المعارف التي تتعلق بالأمور المالية من أسعار، عملات، وطرق دفع. هذا وبالأضافة إلى التعامل مع المعارف المتعلقة بالأماكن الجغرافية من أماكن انطلاق ووصول. كل هذه المعارف ممكن أن تكون في أجزاء بحسب طبيعتها، يمكن لنظام المعلومات في سيناريو نظام رحلات العمل أن يستخدمها لاتخاذ القرارات.

إن الأنطولوجية الجغرافية التي من الممكن استخدامها في نظام معلومات حجز رحلات العمل السابق يجب أن تقوم بترميز معلومات عن القارات، البلدان، الأنهار، الطرقات، خطوط السكك الحديدية والمدن. وعليها أن تقوم بتحديد أماكن الأنهار والمدن وغيرها في المناطق الجغرافية التابعة لها أو التي تقع فيها. بالإضافة إلى ما سبق من تعريف المناطق الجغرافية كمفاهيم عامة وعلاقاتها ببعضها البعض فيجب تعريف هذه المناطق والمدن بأسمائها كأمثلة عن المفاهيم التي تم تعريفها. يمكن توضيح الشبكة الدلالية لهذا المثال بالشكل "2-3".



الشكل 3-2، الشبكة الدلالية لرحلات العمل

نلاحظ في هذا المخطط ما يلي:

- مجموعة من المفاهيم العامة مثل: "بلد"، "قارة"، "منطقة جغرافية"..

- مجموعة من الأمثلة المحددة عن المفاهيم العامة مثل: "أوروبا"، "برلين"..
- علاقة الوراثة "isa" والتي تعرف بـ "Subsumption" والتي تربط بين المفهوم العام والخاص مثل: "River" هو من النوع "BodyofWater" والـ "City" هي من النوع "Infrastructure".
- علاقة "LocatedIn" والتي تحدد علاقة المنطقة الجغرافية الأصغر أو أي غرض أخر بالمفهوم الأكبر.
- علاقة "io" والتي تحدد العلاقة بين المفهوم والأمثلة على هذا المفهوم مثل: "الولايات المتحدة" هي مثال عن المفهوم "بلد"، و "أفريقيا" هي مثال عن المفهوم "قارة" وهكذا.

ليس من السهل تمثيل كل أنواع معلومات الأنطولوجية في شبكة كما في المثال السابق، حيث أنه من غير الممكن تمثيل بعض العلاقات التي من الممكن أن تكون أكثر تعقيداً والتي يفرض فيها قيود على العلاقات بين المفاهيم، لذلك فإن الأدوات التي يستخدمها مهندس الأنطولوجية توفر إمكانيات تحرير متقدمة باستخدام أحد لغات الأنطولوجية.

يوجد العديد من لغات الأنطولوجية والتي تعتمد على طريقة تمثيل المعارف، نعرض في المثال التالي أحد أهم هذه اللغات وهي "OWL" وهي اختصار لـ "Ontology Web Language" والتي تعتمد على الـRDF:


```
...
<owl:Class rdf:ID="City">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#locatedIn"/>
            <owl:allValuesFrom rdf:resource="#PlanarRegion"/>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf rdf:resource="#Infrastructure"/>
    <owl:disjointWith rdf:resource="#Road"/>
    <owl:disjointWith rdf:resource="#IndustrialFacility"/>
</owl:Class>
<owl:ObjectProperty rdf:ID="locatedIn">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:domain rdf:resource="#GeographicLocation"/>
    <rdfs:range rdf:resource="#GeographicRegion"/>
    <owl:inverseOf rdf:resource="#isRegionFor"/>
</owl:ObjectProperty>
<EuropeanCity rdf:ID="London"/>
...
```

إن الترميز السابق يعرض مجموعة المفاهيم كـ "Classes" والعلاقات كـ"Properties" والأمثلة كـ"individuals" باستخدام معرفات الـXML. إن لغة الـOWL وشبيهاتها من لغات الأنطولوجية تعتمد على التمثيل المنطقي "DL". لذلك فإن محرك الاستدلال يتعامل مع الأنطولوجية كمجموعة من التعابير المنطقية التي تحدد العلاقة بين المفاهيم والبديهيات بشكل منطقي "وفقاً لنظرية المنطق". بالتالي يمكنه تحديد إمكانية الوصول إلى استنتاجات جديدة بالاعتماد عليها. كمثال عن التعابير المنطقية "DL" نعرض المثال التالي وفية التعابير الضرورية لتعريف المدينة الاوروبية:

$$
\begin{array}{rcl}
\exists\, locatedIn.\top & \sqsubseteq & GeographicLocation \\
\top & \sqsubseteq & \forall\, locatedIn.GeographicRegion \\
\exists\, contains.\top & \sqsubseteq & GeographicRegion \\
\top & \sqsubseteq & \forall\, contains.GeographicRegion \\
GeographicLocation & \sqsubseteq & =1\; locatedIn \\
Continent & \sqsubseteq & GeographicLocation \\
Continent\;(Europe) & & \\
PlanarRegion & \sqsubseteq & GeographicRegion \\
City & \sqsubseteq & GeographicLocation \sqcap \forall\, locatedIn.PlanarRegion \\
EuropeanCity & \equiv & City \sqcap \forall\, locatedIn.\exists\, contains^{-}.\exists\, locatedIn^{-}.\{Europe\}
\end{array}
$$

إن التعبير الأخير السابق الذي يُعّرف المدينة الأوروبية يمكن كتابته باستخدام منطق الـ "First-Order" بالشكل التالي:

$$
\forall x : (EuropeanCity(x) \leftrightarrow
City(x) \wedge \forall y : (locatedIn(x,y) \rightarrow \exists z : (contains(z,y) \wedge locatedIn(Europe,z))))
$$

والتي يمكن قراءتها نثرياً بالطريقة التالية:

"المدن الأوروبية هي المدن التي تكون كل المناطق الجغرافية الموجودة فيها واقعة في مناطق جغرافية تقع فيها أوروبا ". هذا التمثيل يسمح لمحرك الاستدلال أن يحدد فيما إذا كانت مدينة ما تقع ضمن أوروبا أو لا من خلال الإستنباط ضمن المناطق الجغرافية.

3-2-4. إستخدام الأنطولوجيات:

غالباً ما يتم تمييز الأنطولوجية عن قاعدة المعرفة (Knowledge Base) بأنها يفترض أن توصّف المعرفة على مستوى المخطط العام (Schema Level) من خلال مجموعة من المفاهيم والتعابير العامة، بينما تكون قاعدة المعرفة غنية بالبيانات التي تحتوي أمثلة عن وضع معين. ويمكن اعتبار العلاقة بين الأنطولوجية وقاعدة المعرفة كصلة بين التوصيف المنطقي لمجموعة المعارف الخاصة بمجال ما والأداة التقنية المستخدمة للاستدلال والاستنباط. يمكن من وجهة النظر هذه اعتبار الأنطولوجية كأحد أجزاء المعرفة التي يتم استخدامها من قبل التطبيق المعتمِد على المعارف إلى جانب أجزاء أخرى من المعارف كأنطولوجيات أخرى أو معارف أخرى.

عندما يريد النظام المعتمد على المعارف استشارة الأنطولوجية، فإنه يقوم بتحميل التوصيف الخاص بهذه الأنطولوجية إلى قاعدة معرفته التي يعمل عليها مع المعارف الأخرى التي قد يحتاجها ليتم أخذها بعين الاعتبار أثناء عملية الاستدلال. ففي مثال نظام حجز رحلات العمل والذي سيعتمد على الأنطولوجية الجغرافية، المالية، ومعلومات النقل من أجل المقارنة بين عروض الحجز، سيقوم النظام بتحميل التوصيف الخاص بكل هذه الأنطولوجيات إلى قاعدة المعرفة الخاصة به. بهذه الحالة فإن النظام المعتمد على المعرفة قام باستخدام الأنطولوجية من خلال قاعدة المعرفة الخاصة به.

يمكن استخدام النموذج الخاص بالأنطولوجية بطرق متعددة بناءً على طبيعة التطبيق، سنورد هنا مجموعة من حالات الاستخدام بأشكال متعددة:

- على مستوى الاتصال (Level of knowledge connectivity):

يمكن للتطبيق النظر إلى الأنطولوجية على أنها المصدر الوحيد للمعارف التي يحتاجها ويتصل بها. تستخدم النظم الخبيرة هذه الطريقة للاحتفاظ بقاعدة معرفة عالية التوصيف لاستخدامها للإجابة على الاسئلة الخاصة بمجال ما. وعلى النقيض من ذلك يمكن النظر للأنطولوجية من خلال علاقتها من أنطولوجيات أخرى ومعارف أخرى كمصادر متكاملة للتطبيق لتكوين نموذج واحد. على الأنطولوجية أن تدعم قابلية التشغيل في عدة أنظمة في سيناريو مكاملة المعلومات.

- على مستوى التجريد (Level of knowledge abstraction):

يمكن للتطبيقات إستخدام الأنطولوجية على مستوى المخطط الهيكلي لتصنيف المعرفة. كمثال على هذا التطبيقات التي تقوم بتصنيف المفاهيم المعرفة من قبل المستخدم بشكل اتوماتيكي. من جهة أخرى يمكن استخدام الأنطولوجية نفسها كمخطط هيكلي لمجموعة الأمثلة عن المفاهيم في معارف كبيرة أو قواعد بيانات.

- على مستوى الاتمتة في معالجة المعرفة (Level of automation in knowledge processing):

يمكن للتطبيقات استخدام العديد من تقنيات الاستدلال لاستنباط المعارف الضمنية والكامنة من مجموعة البديهيات المعرفة في الأنطولوجية للاجابة على أسئلة أكثر تعقيداً.

هذا ويمكن في نفس الوقت استخدام الأنطولوجية كمجرد طريقة للتوثيق وليس من أجل الآلة. سنورد في ما يلي مجموعة من الأمثلة عن التطبيقات التي تستخدم النماذج المفاهيمية التي تقدمها الأنطولوجية:

- تكامل المعلومات (Information integration):

يعتبر استخدام الأنطولوجية في المكاملة بين مصادر المعلومات المتنوعة أحد الحقول الواعدة. غالباً ما تحتوي قواعد البيانات المختلفة على نفس نوع المعلومات لكن ضمن نماذج بيانات مختلفة. يمكن في هذه الحالة استخدام الأنطولوجية كوسيط بين بنى قواعج البيانات، مما يسمح بمكاملة المعلومات من مصادر معلومات مختفة وتفسير البيانات من أحدها باستخدام بنية الآخر. ففي مثال نظام حجز رحلات الطيران يمكن استخدام الأنطولوجية الجغرافية للمكاملة بين قواعد البيانات التي تحتوي على معلومات جغرافية لكن بصيغ مختلفة، ففي أحدها يمكن الوصول إلى المدن من خلال علاقتها بالمنطقة الجغرافية التي تقع فيها بينما يتم نمذجتها في قاعدة البيانات الأخرى على أنها منطقة جغرافية يتم تمييزها بأحد الحقول على أنها مدينة أو بلد.

- استرجاع المعلومات (Information retrieval):

يعتبر استخدام الأنطولوجية أحد أهم حقول استخدام الأنطولوجية "إن تجربة شركة غوغل من خلال محرك بحثها في استرجاع المستندات على الوب خير مثال على ذلك". إن الهدف من استخدام الأنطولوجية في استرجاع المعلومات هو زيادة الدقة في استرجاع هذه المعلومات بالاخذ بعين الاعتبار للمعنى الدلالي للمعلومات الموجودة في الاستعلامات والمستندات، بترفيع الكلمات المفتاحية إلى مفاهيم وعلاقات. وكمثال على استخدام الأنطولوجية في استرجاع المعلومات فإن الاستعلام عن عاصمة ألمانيا باللغة الانكليزية "Capital of Germany" والتي تعيد المستندات التي تتحدث عن

برلين، ولأن كلمة "Capital" يمكن أن تتعلق برأس المال والأمور المالية فإن البحث بالكلمات المفتاحية قد يرجع معلومات خاطئة تتحدث عن أمور مالية تتعلق بألمانيا.

- إدارة المحتوى بشكل دلالي (Semantically enhanced content management):

في العديد من التطبيقات يتم استخدام معرفات البيانات "Meta data" لاستخدام البيانات. إن الأنطولوجية يمكنها تزويدنا بالمصطلحات الخاصة بمجال محدد للإشارة لمعرفات البيانات هذه. إن استخدام التمثيل الصوري من قبل لغات الأنطولوجية يسمح بالمعالجة المؤتمتة لمعرفات البيانات هذه مع إضافة إمكانية قابلية التشغيل المتعدد بين الآلات. فعلى سبيل المثال يمكن استخدام الأنطولوجية الجغرافية للإشارة إلى المحتوى الجغرافي للعديد من المصادر كالكتب الجغرافية، والمقالات في المكتبات الالكترونية بالتالي إمكانية الوصول إليها وأرشفتها بشكل أفضل.

- إدارة المعرفة ونوافذ المجتمعات (Knowledge management and community portals):

يتم النظر في الشركات أو المجتمعات الاخرى المنظمة إلى المعارف الفردية كمصادر استراتيجية التي يمكن ان توصف من أجل المشاركة، يعرف هذا بإدارة المعرفة. ويمكن للأنطولوجية أن تزودنا بطريقة لإدارة المعرفة ضمن نموذج مفاهيمي مشترك للربط بين الأنظمة التقنية لاستعراض، لحفظ، للبحث وتبادل المعرفة.

- الأنظمة الخبيرة (Expert systems):

هناك رغبة في العديد من المجالات، كالتشخيص الطبي أو الاستشارة القانونية بمحاكاة عمل خبير هذه المجالات للاجابة عن أسئلة واستفسارات معقدة. يتم ذلك في النظم الخبيرة من خلال التعاون بين العديد من الأنطولوجيات في مجال ما لتوصيف المعارف عندها ومن خلال الاستدلال عبر هذه المعارف يمكن الإجابة عن أسئلة معقدة مثل: "ما هي المدينة الألمانية الأقرب إلى الحدود الفرنسية" أو "ما هي المدن التي يعبّر فيها نهر الراين".

3-2-5. أهداف الأنطولوجية:

1- تحدد طبيعة الواقع أو تعرّفها بتحديد المفاهيم والكيانات والمصطلحات والفئات في ميدان معين من أجل نمذجة العلاقات بينها أو صياغتها.

2- نشأت لجعل الغموض المفاهيمي والدلالي في حده الأدنى في بيئة معلوماتية وتكنولوجية.

3- مفيدة في تعزيز قابلية التشغيل التبادلي بين الأنظمة في الميادين المعرفية المختلفة، أو لإنشاء الوكلاء الأذكياء الذين يمكنهم إنجاز أعمال معينة، وكلاهما من أغراض الوب الدلالي.

4- إن الأنطولوجية التي تعمل من أجل تنظيم الوثائق واسترجاعها ينبغي أن تصيغ واقع استخدام اللغة من أجل الاتصال أو تشكله. وهذه يطلق عليها أحياناً الأنطولوجية اللغوية، وتشمل مسائل النحو، والدلالة، وبناء الجملة، وما شابه.

3-2-6. أنواع الأنطولوجية:

منذ بدايات الدراسات المتعلقة بالأنطولوجية في علوم الحاسبات تم النظر إلى الأنطولوجية كطريقة يمكن إعادة استخدامها في هندسة النظم المعتمدة على المعارف (Knowledge-based systems)، على أن تتم إعادة استخدام الأنطولوجية حسب نوعها في المجال الذي توصفه. يمكن تصنيف الأنطولوجيات تبعاً لعدة مقاييس والتي تتراوح بين مستوى تعميم المفاهيم التي توصفها ونمط المعرفة التي تنمذجها. سنستعرض التصنيفات الأكثر استخداماً للأنطولوجيات.

المقياس الأول للتصنيف هو مستوى عمومية المفاهيم المستخدمة لتوصيف المجال المدروس، وهنا نستطيع تمييز الأنطولوجيات التالية [Guarino 1998]:

1- أنطولوجية المستوى الأول (top-level ontology):
تصف هذه الأنطولوجيات مفاهيم عامة ومعرفة شائعة كالفضاء والوقت والأحداث (events).

2- أنطولوجية المجال (domain ontology):
تصف المفردات المتعلقة بمجال معين كالطب او الفيزياء.

3- أنطولوجية المهام (task ontology):
تصف المفردات المتعلقة بمهمة أو فعالية ما كالتشخيص أو البيع.

4- أنطولوجية التطبيقات (application ontology):
تصف المفردات المتعلقة بمجال معين وبمهمة ما تابعة لهذا المجال كالتشخيص الطبي مثلاً. نلاحظ أن هذه الأنطولوجيات تجمع بين النوعين السابقن من الأنطولوجيات.

اقترح Van Heijst وزملاؤه [Heijst 97] تصنيف الأنطولوجيات تبعاً لمقياسين، المقياس الأول هو نمط المفردات الموضوعة في الأنطولوجية، وهو يصنف الأنطولوجيات إلى:

1- أنطولوجيات المصطلحات (Terminological Ontologies):
وهي مجرد قاموس يحوي الصطلحات المستخدمة لتمثيل المعرفة في مجال ما، إلا أنها لا تحوي تعريفاً دقيقاً لمعنى كلٍ من هذه المصطلحات.

2- أنطولوجيات المعلومات (Information Ontologies):
وهي تحدد بنية التسجيلة في قواعد المعطيات إلا أنها لا تحدد المفاهيم التي ستوضع في هذه البنية.

3- أنطولوجيات نمذجة المعرفة (Knowledge Modeling Ontology):
وهي تحدد المفاهيم المتعلقة بمعرفةٍ ما. إنها أغنى من أنطولوجيات المعلومات وغالباً ما تهتم بمجال استخدام معين للمعلرفة التي تنمذجها.

أما المقياس الثاني فهو الموضوع المطلوب نمذجته من الأنطولوجية، وهنا نجد الأنواع التالية:

1- أنطولوجيات التطبيقات (Application Ontologies):
وتحدد المفاهيم الضرورية لنمذجة المعرفة المطلوبة من قبل تطبيق معين. هذه الأنطولوجيات غير قابلة لإعادة الاستخدام مباشرةً من قبل تطبيقات أخرى، بل تحتاج إلى إعادة هيكلة وصياغة كي تصبح قادرة على العمل مع تلك التطبيقات.

2- أنطولوجيات المجال (Domain Ontologies):
وتحدد المفاهيم الخاصة بمجال معين.

3- الأنطولوجيات العمومية (Generic Ontologies):
وتحدد المفاهيم العامة المشتركة بين عدة مجالات، وهي تتوافق مع الأنطولوجيات العالية المستوى التي ذكرناها سابقاً.

يمكن أن تختلف الأنطولوجيات أيضاً بمقدار التجريد المستخدم في توصيف المفاهيم ومعانيهم، أي يمكن أن نجد أنطولوجيةتين حول الموضوع نفسه لكنهما تختلفان في طريقة توصيف مفردات هذا المفهوم [Uschold 1996]. وهنا نميز بين الأنطولوجيات: ضعيفة التجريد (high informal)، شبه ضعيفة التجريد (semi-informal)، شبه قوية التجريد (semi-formal)، وقوية التجريد (rigorously-formal).

3-2-7. تكامل الأنطولوجية:

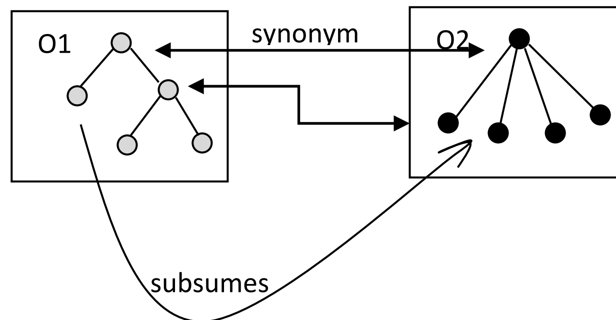
لو نظرنا إلى الوب في الوقت الحالي فسنرى أنه يحوي الكثير والكثير من مصادر المعلومات حول العديد من الموضوعات، ولو دققنا النظر لوجدنا أن الكثير من هذه المصادر تحوي معلومات متداخلة (Overlapping Information) حول الموضوع نفسه. لو فرضنا أننا نرغب بمعرفة معلومات حول موضوع ما، فسيكون الأمر عسيراً جداً على أي تطبيق أن يأتي بهذه المعلومات من تلك المصادر الموزعة على الوب كل على حدا لهذا السبب كان الهدف من تكامل النطولوجيات هو توحيد المعلومات بشكل ذكي والتخلص من التكرار الموجود في العديد من المصادر كي تتمكن التطبيقات من جلب المعلومات المطلوبة وتمكين المستخدم من الإطلاع على هذه المعلومات دون الحاجة إلى أن يعرف أي شيء عن البنية التحتية لها.

3-2-7-1- طرق تكامل الأنطولوجيات:

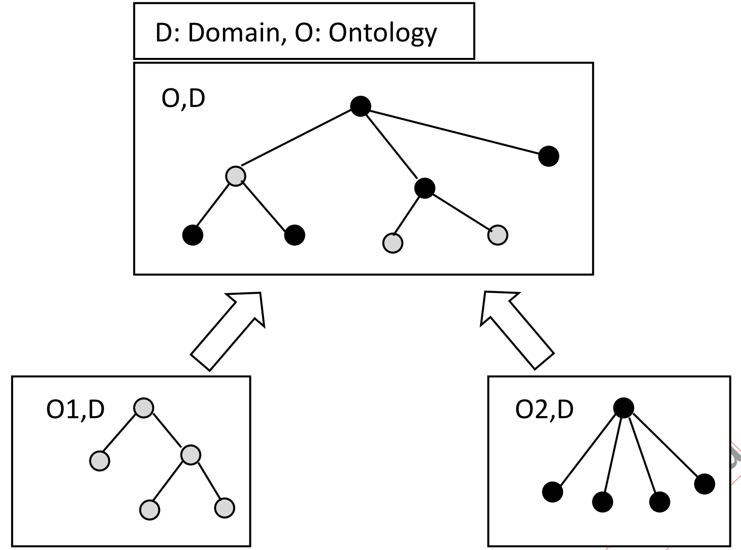
كما ورد في [Jakoniene 2003]، فإن هناك أربعة طرق تمكننا من توحيد الأنطولوجيات هي:

1- التقابل (Mapping): وهنا يتم ربط المفاهيم أو العلاقات الموجودة في الأنطولوجيات المختلفة بعلاقة مساواة (Equivalency).

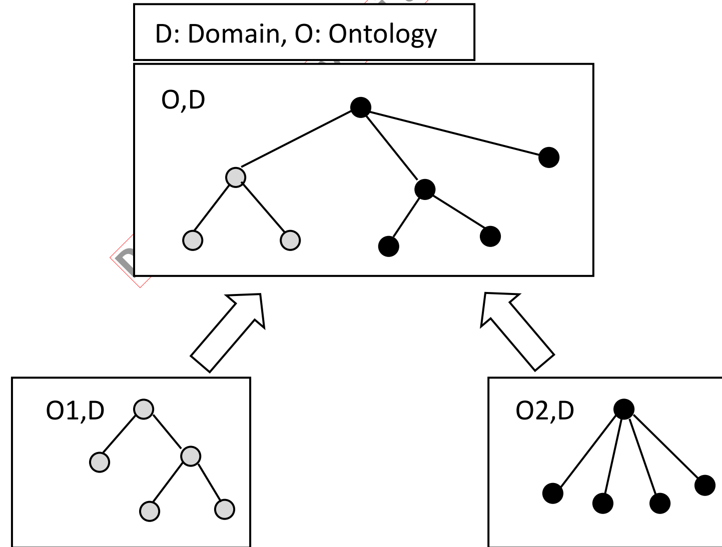
2- التراصف (Aligning): وهنا يتم الاتفاق بين أنطولوجيةتين او أكثر حول ربط المفاهيم والعلاقات الخاصة بها بعلاقات معينة (مثل علاقة الترادف synonym relation). انظر الشكل التالي:



3- الدمج (Merging): وهنا يتم دمج عدة أنطولوجيات مختصة بموضوع واحد لتنتج أنطولوجية جديدة توحد المعارف الموجودة في كل من تلك الأنطولوجيات. انظر الشكل التالي:



4- التكامل (Integration): وهنا نكامل عدة أنطولوجيات مختصة بعدة مواضيع ويتم بناء أنطولوجية جديدة تمثل جميع تلك الأنطولوجيات. انظر الشكل التالي:



3-2-7-2- خطوات التكامل:

كي تتم عملية المكاملة علينا مراعاة الخطوات التالي كما هي في [Maier 2003]:

1- تحديد جميع الأنطولوجيات المطلوب مكاملتها؛
2- مقابلة ودمج (Map and Merge) هذه الأنطولوجيات؛
3- تعريف المنطق (Logic) وقواعد الاستدلال الجديدة للأنطولوجية الناتجة؛
4- توفير مكان (Data storage) لتخزين الأنطولوجية الناتجة.

3-2-7-3- معوقات التكامل:

إن تكامل الأنطولوجيات من العمليات الصعبة جداً، إذ هناك العديد من المعوقات التي تظهر عندما نبدأ بها. من هذه المعوقات الاختلافات بين الأنطولوجيات على عدة مستويات. في [Jakoniene 2003] نجد:

- الاختلافات على مستوى اللغة: حيث يمكن أن تكون كل أنطولوجية مكتوبة بلغة مختلفة.
- الاختلافات على مستوى الأنطولوجية نفسها: فكل أنطولوجية – كما مر معنا – يمكن ان تكتب بطريقة خاصة حتى ولو كانت الأنطولوجيات تتناول الموضوع نفسه. من هذه الاختلافات نجد الاختلافات في المصطلحات (Car ،Automobile مثلاً) أو الاختلافات في الترميز (-mm dd/mm/yyyy،dd-yy مثلاً).

3-3. الوب الدلالي:

تعتبر شبكة الوب أغنى المصادر المعلوماتية بما تحويه من مستندات ومعلومات ومصادر منوعة يمكن الوصول إليها عن طريق محركات البحث التقليدية. غير أن تنظيم هذه المعلومات والمستندات بصورة تسهل عملية البحث فيها والوصول إليها، يعتبر أمراً غاية في الصعوبة. يضاف إلى ذلك، أنه في ظل التزايد المستمر في حجم المعلومات المنشورة في شبكة الوب أصبح من الصعوبة بمكان قيام محركات البحث بإيجاد المعلومات المناسبة. ومن هذه المشكلة ظهرت فكرة "الوب ذات الدلالات والمعاني اللفظية"، أو ما يطلق عليه مصطلح (Semantic Web). وقد لخص Tim Berners Lee التطبيقات العملية للوب الدلالي قائلاً: أحلم بشبكة وب تصبح فيها الحاسبات قادرة على تحليل كل البيانات على شبكة الوب من المحتوى والروابط والعمليات بين الناس والحاسبات. ولكن الوب الدلالي الذي سيجعل هذا ممكناً لم يظهر بعد، ولكنه عندما يظهر، سيتم التعامل مع الآليات اليومية للتجارة والحياة اليومية بواسطة الآلات التي تتحدث مع آلات، ويمكننا أن نقول أخيراً إننا نحيا عصر الوب الذكي.

3-3-1. تعاريف:

نورد هنا مجموعة من التعاريف للوب الدلالي:

- الوب الدلالي: هو ثورة جديدة في عالم الوب حيث تصبح المعلومات والبيانات قابلة للمعالجة منطقياً من قبل برامج الحاسوب بحيث تتحول تلك المعلومات والبيانات إلى شبكة بيانات ذات معنىً، أي أنه يمكن للبرامج الحاسوبية الخاصة أن تعرف ماذا تعني هذه البيانات.
- الوب الدلالي: مجموعة من الطرائق والتقنيات المتبعة لجعل الآلات قادرة على فهم المعاني أو الدلالات للمعلومات على الشبكة العنكبوتية العالمية.
- الوب الدلالي: شبكة من البيانات التي يمكن معالجتها من قبل الآلات بشكل مباشر أو غير مباشر.
- الوب الدلالي: هو جعل الوب أكثر قابلية للفهم من قبل الآلات.
- الوب الدلالي: هو بناء بنية تحتية مناسبة للعملاء الأذكياء (Intelligent Agents) للقيام بعمليات معقّدة بالنيابة عن مستخدميهم.
- الوب الدلالي: هو التعريف الصريح عن المعلومات الموجودة في العديد من تطبيقات الوب، ودمج المعلومات بطريقة ذكية، وتوفير الوصول الدلالي إلى الانترنت، واستخراج المعرفة من النصوص.

2-3-3. سمات عامة لمحتوى الوب الدلالي:

يسعى الخبراء والمتخصصون إلى الإتفاق على مجموعة من السمات العامة للمحتوى المعد للعمل مع الوب الدلالي، وتتضمن هذه السمات أن يكون المحتوى متضمناً:

1- مستندات مميزة بمعلومات دلالية (امتداد لبطاقات HTML المستخدمة فى صفحات الوب الحالية لتوفير المعلومات لمحركات البحث التى تستخدم أدوات تقصّي الوب)، وهذه المعلومات قد تكون معلومات عن محتوى ثقافي وعلمي بشري مثل المؤلف والعنوان والوصف وتستطيع البرامج قراءتها وفهمها، أو قد تكون معلومات وصفية خالصة تمثل مجموعة من الحقائق مثل الموارد والخدمات الموجودة فى الموقع.

2- ألفاظ موحدة لبيانات الوصف وهي مراجع وصف المعرفات والعلاقات بينها(الأنطولوجيات)، وتتيح المراجع لمصممي المستندات معرفة كيف يقومون بترميز مستنداتهم بحيث تستطيع برامج البحث والتصفح استخدام المعلومات فى بيانات الوصف المقدمة (بحيث لا يتم الخلط بين معنى مؤلف صفحة الوب ومعنى مؤلف الكتاب موضوع المراجعة).

3- العمل مع برامج بحث وتصفح دلالية آلية تقوم بإنجاز المهام لمستخدمي الوب الدلالي باستخدام هذه البيانات.

4- القدرة على تقديم خدمات تعتمد على الوب (عادةً ما تكون مزودة ببرامج بحث خاصة) لتوفير المعلومات لبرامج البحث بشكل خاص (على سبيل المثال خدمة ثقة يطلبها برنامج بحث يستفسر ما إذا كان تاريخ شركة تجارة إلكترونية معينة على الإنترنت يتضمن تقديم خدمات ضعيفة أو إرسال بريد دعائي مزعج).

3-3-3. التحديات التي تواجه الوب الدلالي:

رغم المزايا الكبيرة التي يبشّر بها الوب الدلالي، فإن نجاحه أو فشله يتوقف على الوصول السهل وتوفر محتوى متنوع يمتاز بالتميّز والجودة العالية. وما زالت هناك عدة قضايا تواجه الوب الدلالي ومنها الضخامة، والغموض، وعدم التأكد، والتناقض، والخداع وغيرها. وتقع على عاتق الأنظمة المؤتمتة مهمة التعامل مع هذه القضايا للوصول إلى الهدف من الوب الدلالي.

1. الضخامة:

تحتوي الشبكة العنكبوتية العالمية على الأقل على 14.29 بليون صفحة، لذا فأنظمة الأتمتة يجب أن تكون قادرة على التعامل مع إدخالات كبيرة جداً.

2. الغموض:

يوجد بعض المفاهيم غير الدقيقة مثل (طويل) أو (مسن) حيث لا يمكننا الحكم على شي بأحد هاتين الصفتين. وتأتي هذه المشكلة من اختلاف معنى هذه الصفات من منظور لآخر. المنطق الغامض (Fuzzy Logic) هو التقنية الأكثر انتشاراً للتعامل مع مشاكل الغموض.

3. عدم التأكد:

يأتي عدم التأكد من وجود مفاهيم دقيقة لكن قيمها غير دقيقة، مثلاً يمكن لمريض أن يملك عدداً من الأعراض التي تؤشر إلى أكثر من مرض وباحتمالية مختلفة، وتقنيات المنطق الاحتمالي توظف في حل مثل هذه المشاكل.

4. التناقض:
لا بد أن تظهر التناقضات المنطقية من نمو الأنطولوجيات الكبيرة، وعندما يتم جمع أنطولوجيات من مصادر مختلفة. المنطق الاستنتاجي يفشل بشكل كارثي عند يواجه التناقض. المنطق القابل للإبطال هو أحد طرق حل هذه المشكلة.
5. الخداع:
ويحصل عندما يتعمّد مقدّم المعلومات تضليل المستهلك. يستفاد من تقنيات التشفير للتخفيف من هذه المشكلة.
6. توفر المحتوى:
فى الوقت الحالي، لا يوجد سوى القليل من محتوى الوب الدلالي، لهذا ينبغي ترقية محتوى الوب الحالي إلى محتوى الوب الدلالي بما فى ذلك صفحات HTML الثابتة ومحتوى XML الحالي والمحتوى التفاعلي والوسائط المتعددة وخدمات الوب.
7. إنشاء مراجع وصف المعرفات وتوفرها وتطويرها:
فمرجع الوصف سيصبح جزءًا أساسيًا فى الوب الدلالي لأنه هو الذى يوضح العلاقات الدلالية بين محتوى الوب، ولهذا يجب بذل مجهود كبير فى إنشاء مرجع وصف شائع الاستخدام للوب الدلالي، بشرط توافر البنية الأساسية الكافية لتطوير مجموعات الوصف وإدارة وتوزيع التغيير فيها.
8. القدرة على التوسّع:
يجب بذل جهود كبيرة لتنظيم محتوى الوب الدلالي وتخزينه وتوفير الآليات الضرورية للعثور عليه، ويجب تنفيذ كل هذه المهام وتنسيقها بشكل متدرّج، حيث ينبغي تجهيز هذه الحلول للنمو الضخم للوب الدلالي.
9. تعدد اللغات:
توجد هذه المشكلة فى الوب الحالي، وينبغي حسمها فى الوب الدلالي، فأي منهج للتعامل مع الوب الدلالي ينبغي أن يوفر الأدوات أو الوسائل للوصول إلى المعلومات بلغات عديدة، مما يتيح إنشاء محتوى الوب الدلالي والوصول إليه بشكل مستقل عن اللغة الأصلية لموفّري المحتوى ومستخدميه.
10. طريقة العرض:
ستصبح طريقة العرض البديهية لمحتوى الوب الدلالي أمراً ملحّاً لحل مشكلة الفيض الهائل من المعلومات، لأن المستخدمين سيطالبون بالتعرف السهل على المحتوى المطلوب للأغراض الخاصة بهم، ولهذا يجب اكتشاف أساليب جديدة تختلف عن طريقة العرض فى الوب الحالية القائمة على بنية النص التشعبي.
11. استقرار لغات الوب الدلالية:
فى النهاية يجب إجراء جهود توحيد المعايير بشكل ملح في هذا المجال الواعد، للسماح بإنشاء التكنولوجيا الضرورية التي تدعم الوب الدلالي.

4-3-3. مشاريع في الوب الدلالي

هناك بعض المواقع والجهات التى بدأت بالفعل بتطبيق مفاهيم الوب الدلالية عملياً، سواء في تجهيز المحتوى أو البحث فيه، ومن التطبيقات الشهيرة للوب الدلالي الحالية:

- DBPedia:

يهدف هذا المشروع إلى استخراج المعلومات المنظّمة من المعلومات الموجودة على موقع الويكيبيديا، ومن ثم عرض هذه المعلومات على الوب. يسمح هذا المشروع بالإستعلام عن الخصائص والعلاقات المرتبطة بموارد المعلومات الموجودة على الويكيبيديا بالإضافة إلى وصلات لقواعد بيانات أخرى

مرتبطة. يصف Tim Berners Lee هذا المشروع بأنه أحد أشهر أجزاء مشروع "LinkedData".

- Friend of a Friend (FOAF):

يعتبر هذا المشروع أول مشروع في الوب الدلالي الاجتماعي حيث أنه يجمع بين تقنية الـ RDF واهتمامات الوب الاجتماعي وهو أنطولوجية تفهمها الحواسيب تصف الأشخاص، وأنشطتهم وعلاقتهم مع الأشخاص الآخرين والأشياء، ويسمح هذا المشروع بوصف الشبكات الاجتماعية (Social Networks) دون الحاجة إلى قاعدة بيانات مركزية. كتبت أنطولوجية المشروع باستخدم اللغتين RDF وOWL ويمكن أن تستخدم الحواسيب هذه الأنطولوجية فمثلاً للحصول على قائمة بأسماء جميع الأشخاص الذين يقطنون في أوروبا أو قائمة بأسماء الأشخاص الذين يعرفهم صديقك وتعرفهم أنت.

- GoodRelations:

هذا المشروع هو أنطولوجية مشهورة في التجارة الإلكترونية وتستخدم في التعبير عن معلومات المنتجات، والأسعار، وطرق الدفع وغيرها. ومن الشركات التي تبنت هذه الأنطولوجية في تعاملاتها التجارية (Google،BestBuy ،Overstock ،Yahoo ،OpenLink Software ، O'Reilly Media،the Book Mashup) وغيرها.

- SIOC Semantically-Interlinked Online Communities:

يوفر هذا المشروع أنطولوجية توضح مفردات لمصطلحات وعلاقات تمثل فضاءات بيانات الوب. وتمثل فضاءات بيانات الوب كل من منتديات المناقشة والمدونات والقوائم البريدية ومعارض الصور وغيرها.

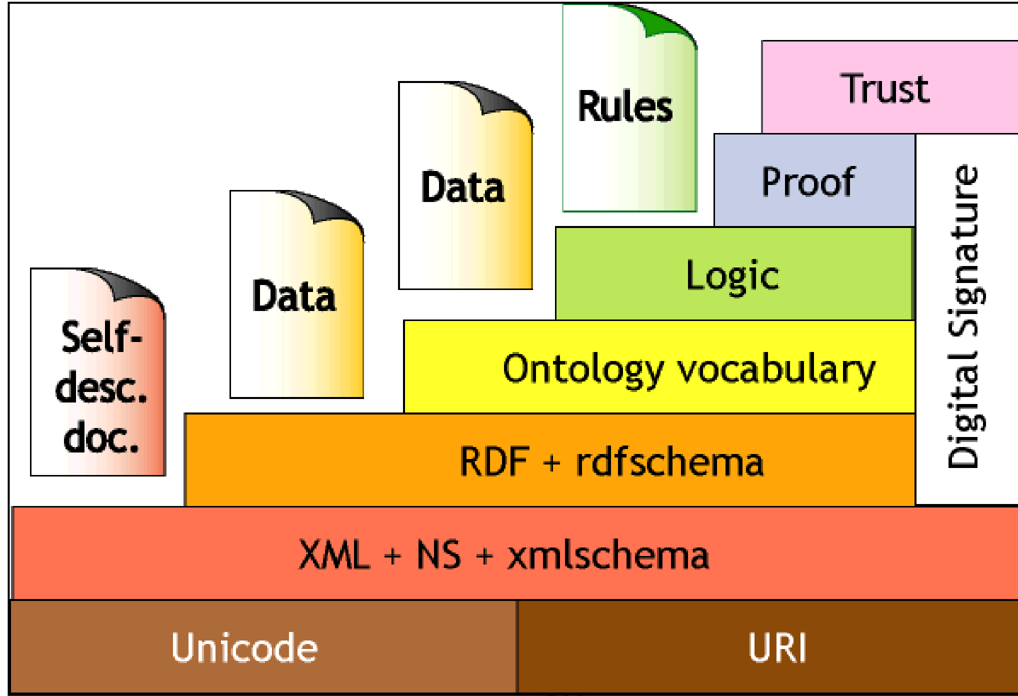
3-4. الأنطولوجية في الوب الدلالي:

إذا كان الوب الدلالي يتطلب أن تكون المصطلحات لها معنى واضح حتى يمكن للآلات أن تعالج المعلومات الموجودة على الوب أو تجهزها أوتوماتيكياً؛ فإن الأنطولوجيات هي التي تقدم مثل هذا المعنى، وهي كتل البناء للوب الدلالي لتستخدم مع XML و RDF. والأنطولوجيات مهمة إذا كان الوكلاء على الوب قادرين على البحث و/ أو الدمج للمعلومات من تجمّعات متنوعة، وذلك لأن المصطلح نفسه قد يستخدم في سياقات مختلفة بمعانٍ مختلفة، وأن المعنى نفسه قد يمثل بمصطلحات مختلفة في سياقات مختلفة.

3-4-1. لغات الأنطولوجية:

يعتمد الوب الحالي على لغة HTML التي تحدد كيفية تخطيط صفحة الوب للقارئ البشري، ولكن لغة HTML بوضعها الحالي لا يمكن استغلالها بواسطة تقنيات استرجاع المعلومات لتحسين النتائج التي طالما اعتمدت على الكلمات الأساسية التي تشكل نص ومحتوى الصفحة، ومن ثم أصبحت مقيدة بكلمات البحث. فى الوب الدلالي، لا تخزن الصفحات المحتوى كمجموعة من الكلمات والألفاظ غير المترابطة فى مستند معين، ولكنها تقوم بترميز المعنى والتركيب. بصفة عامة تعتمد لغات الوب الدلالية على لغة XML ثم تصعد هرم اللغات لتصل إلى لغة RDF وRDFS وOIL وDAML+OIL وOWL. هذه اللغات أكثر ثراءً وتعقيداً من لغة HTML وتتيح إلى حد كبير تمثيل المعنى وبنية المحتوى (العلاقات الداخلية بين المفاهيم)، وهذا يسهم فى جعل محتوى الوب مفهوماً

وقابلاً للمعالجة بواسطة برامج البحث الدلالي، مما يفتح الطريق واسعاً أمام جيل جديد من التقنيات والخدمات المبتكرة من خلال الوب ونماذج الأعمال، حيث ننتقل تدريجياً من دعم استرجاع المعلومات إلى تفويض المهام وإنجازها.



3-4-1-1. RDF(S):

تعرف لغة توصيف الموارد (RDF – Resource Description Framework) [Klyne،Carroll 2004] بأنها أحد معايير الـW3C لتمثيل المعلومات الخاصة بالموارد على الشبكة العنكبوتية. وتهدف بشكل أساسي لتمثيل معلومات البيانات (meta data) الخاصة بالموارد على الوب كعنوان ومؤلف صفحة الوب، موضوع مقالة الكترونية، أو وظيفة أحد خدمات الوب.

أما الـRDFS "RDF Schema" فهي توسعة للـRDF لتسهيل عملية تمثيل مصطلحات الـRDF نفسها ويمكن القول بأنها توصيف التوصيف (RDF of RDF) [Brickley،Guha 2004] . في حين أن الـRDF يقوم بربط الموارد بخصائص فإن الـRDFS يقدّم الفائات التي تنتمي إليها هذه الموارد وهرميتها.

إن الاستخدام المشترك لكل من الـRDF والـRDFS يشار إليه بـRDF(S).

3-4-1-1-1. العناصر الأساسية في الـRDF:

- معرّف الموارد (URI):

يتم استخدام عناوين URI لتسمية الكينونات على الوب. وهي عبارة عن طريقة تسمية تسمح بتقسيم الأسماء إلى عدة أجزاء تمثل فضاءات العمل (namespaces). يتم في الـRDF استخدام هذه العناوين لتعريف أو تسمية أنواع مختلفة من الكينونات على الوب: كتسمية الشيء نفسه "الشركة س" أو "الموظف ص"، كما ويمكن أن تستخدم لتسمية أنواع الأشياء مثل "شركة" أو "موظف"، أو خصائص لهذه الأشياء كالبريد الالكتروني "mailto:salah@Damascus.com".

باستخدام الـURI يمكن تسمية الكينونات والموارد على الوب بشكل فريد وغير متكرر.

- الجمل الثلاثية (Subject،Predicate and Object):

يتم في الـRDF استخدام جمل ثلاثية تسمى بثلاثيات الـRDF وهي بالشكل التالي:



حيث يتم تحديد مكونات هذه الجملة باستخدام عناوين "URIs" الموارد. حيث يمثل الـSubject المورد الذي نريد توصيفه، والـPredicate يمثل خاصية محددة من خواص هذا المورد، والـObject فيمثل قيمة هذه الخاصية لهذا المورد.

مثال 1: إن "MrX" موظف لدى الشركة "UbiqBiz".



مثال 2: إن "Mrx" هو مؤلف صفحة الوب الخاصة به في موقع "UbiqBiz".



- التمثيل بشكل بيان (Graph Representation):

يتم توليد الجمل الثلاثي في الـRDF من خلال بيان الـRDF (RDF graph)، حيث تمثل عقده عناوين الموارد و تمثل أقواسه الخصائص.

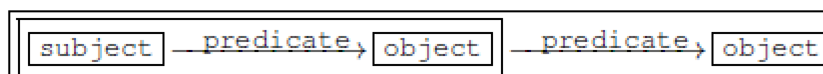
- XML:

من الممكن استخدام الـXML لترميز بيان الـRDF بطريقة قابلة للمعاجة من قبل الآلة وللنقل عبر الشبكة. يظهر المثال السابق بالشكل التالي:

```
<rdf:Description rdf:about="http://ubiqbiz.com/web/MrX.html">
     <btr:hasAuthor rdf:resource="btr:MisterX"/>
</rdf:Description>
<rdf:Description rdf:about="btr:MisterX">
     <btr:employedAt rdf:resource="btr:UbiqBiz"/>
</rdf:Description>
```

- تعددية الجمل الثلاثية (Reification):

يسمح الـRDF باستخدام الجمل الثلاثية ضمن الجمل الثلاثية، حيث تعتبر الجملة الأولى بمثابة المورد الذي يمثل حدوث أو وقوع ثلاثية الـRDF. وتعتبر هذه الخاصية من الخواص المهمة في الوب الدلالي. وتظهر بالشكل التالي:



3-4-1-1-2. استخدام الـRDF(S):

يقوم الـRDFS بتسهيل توصيف مصطلحات الأنطولوجية الخاصة بالتطبيق من خلال استخدام مجموعة من الكلمات المحجوزة التي تستخدم للتعبير عن الفئات (Classes) التي تنتمي إليها الموارد.

- الفئات (Classes):

يستخدم الـRDFS مفهوم الفئة أو الصنف للتعبير عن نوع مورد الـRDF. كما ويستخدم مفهوم الـsubClassOf للتعبير عن العلاقة الجزئية لفئة ضمن فئة أخرى. ويتم تمثيل ذلك في بيان الـRDF(S) بالشكل التالي:

btr:MrX —rdf:type→ btr:Employee —rdfs:subClassOf→ btr:Person

أي أن "MrX" من النوع "موظف" أو بمعنى آخر أن "Mrx" ينتمي لفئة "الموظفين" وهذه الفئة بدورها تعتبر فئة جزئية من الفئة "شخص".

- الخصائص (Properties):

وفقاً للـRDF(S) فإن أي مورد يتم استخدامه كـPredicate ينتمي للفئة "rdfs:Property". وكما بالنسبة للفئات يمكن تنظيم الخصائص ضمن هرمية باستخدام الكلمة "rdfs:subPropertyOf".

btr:employedAt —rdfs:subPropertyOf→ btr:worksFor

ويعبّر هذا المثال على أن كل شخص موظف عند شركة فهو يعمل لدى هذه الشركة.

كما ويمكن من خلال استخدام "rdfs:domain" و "rdfs:range" فيمكن تعريف خاصية ميدان العمل ومجال هذه الخاصية.

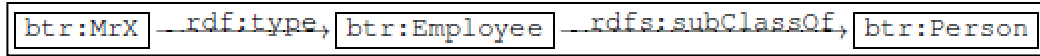
3-4-1-1-3. البعد الدلالي في الـRDF(S):

سنعرض بعض الأفكار التي تظهر المعنى الدلالي للـRDF(S) مما يضفي عليها إمكانيات الاستدلال.

يتم توصيف الـRDF(S) على شكل مجموعة من الثلاثيات والقواعد لاستنتاج واستدلال ثلاثيات جديدة. من خلال المثال التالي أجزاء هذه القواعد في البيان G_{RDF} :

(1)	IF G_{RDF} contains (C, rdfs:subClassOf, D) and (R, rdf:type, C) THEN derive (R, rdf:type, D)
(2)	IF G_{RDF} contains (C, rdfs:subClassOf, D) and (D, rdfs:subClassOf, E) THEN derive (C, rdfs:subClassOf, E)

حيث توصّف القاعدة الأولى مفهوم الفئة الأب (superclass) أما القاعدة الثانية فتعبر عن تعددية علاقة الفئات الجزئية (subclass). يمكن من خلال الثلاثية التالية الأولى استنتاج الثلاثية التالية الثانية:





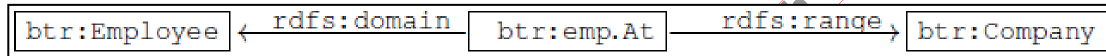
الثلاثية التي تم استنتاجها

بهذا الشكل فإن "MrX" سيتم إيجاده في نظام الـRDF عند السؤال عن قائمة كل الأشخاص. وفي مثال آخر لخصائص الميدان والمجال نعرض القواعد التالية:

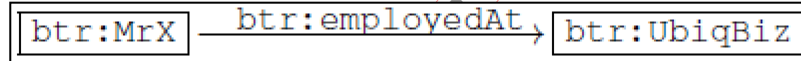
(3) IF G_{RDF} contains (P, `rdfs:domain`, C) and (R, P, S)
THEN derive (R, `rdf:type`, C)

(4) IF G_{RDF} contains (P, `rdfs:range`, C) and (R, P, S)
THEN derive (S, `rdf:type`, C)

باستخدام المواصفات التالية "btr:employedAt"، "btr:Employee"، و"btr:Company":

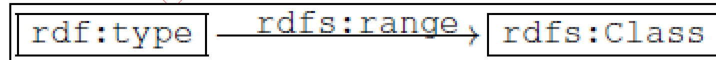


يمكن استنتاج الثلاثية التالية:



الثلاثية التي تم استنتاجها

والتي تقول بأن "MrX" هو موظف لدى الشركة "UbiqBiz" ويمكن أيضاً بنفس الطريقة استخدام القواعد لتحديد العلاقات بين مصطلحات الموارد كما في المثال التالي:



3-4-1-2. OWL:

وهي لغة أنطولوجية للوب تم طرحها كلغة معيارية للتوصيف الدلالي لمحتوى الوب من قبل هيئة الـW3C [Schneider، Hayes، Horrocks 2002] .

تأتي هذه اللغة بثلاثة أنواع تختلف بمدى تصريحية وتعبيرية اللغة وهي:

- OWL-Lite
- OWL-DL
- OWL-Full

تعتمد كل من الـOWL-Lite،OWL-DL على المنطق الوصفي (description logics) لذلك تتوافق مع $\mathcal{SHIF}(\mathcal{D})$ و$\mathcal{SHOIN}(\mathcal{D})$. أما الـOWL-Full فتعتمد على RDF(S).

3-4-1-2-1. قواعد اللغة:

تعرّف هذه اللغة المعيارية عدة أنواع من التعبير بالاعتماد على الـRDF(S)، XML، والشكل النصي. حيث تسمح اللغة المعتمدة على الـRDF بترميز الأنطولوجية من خلال إطار عمل الـRDF(S)، أما بالاعتماد على XML فيتم ترميز الأنطولوجية باستخدام الـXML بشكل مستقل عن الـRDF(S) وتتميز هاتين الطريقتين بأنها موجهة للآلة. أما الشكل الثالث الذي من الممكن أن تأتي به هذه اللغة فهو الشكل النصي الذي يشكل طريقة قابلة للقراءة البشرية موجّهة لمهندس المعارف. كما ويمكن تقديم اللغة بطريقة علمية قابلة للقراءة عن طريق استخدام صيغ المنطق الوصفي (DL formulas).

بشكل مشابه للـRDF(S) فإن لغة الـOWL بنى نمذجة للعناصر الأساسية للأنطولوجية. كالمفاهيم، العلاقات والأمثلة(*concepts*، *relations* and *instances*). ويطلق عليهم في الـOWL على التوالي: صفوف، خصائص، وأفراد(*classes*، *properties*، *individuals* and). أما في المنطق الوصفي فيشار إليها على التوالي: مفاهيم، دور، أفراد(*concepts*، *role*، and *individuals*).

تمتاز الـOWL-DL بإمكانية بناء فئات أكثر تعقيداً باستخدام البواني (*constructors*). سنعرض ذلك في الأمثلة التالية المأخوذة من الأنطولوجية الجغرافية الموضحة في الشكل "3-2".

3-4-1-2-2. أمثلة عن الـOWL:

سيتم في الأمثلة التالية عرض التوصيف النصي لكل مثال إلى جانب عبارات الـOWL في الجانب الأيسر من كل مثال، بالإضافة إلى صيغ الـDL على الجانب الأيمن من كل مثال.

للتصريح عن صف أو فئة (class) جديدة يتم استخدام كلمة *class* كما في المثال التالي:

① "*A continent is a geographic location different from a country.*"	
Class(*Continent* partial intersectionOf(*GeographicLocation* complementOf(*Country*))	$Continent \sqsubseteq GeographicLocation \sqcap \neg Country$

في المثال السابق تم التصريح عن فئة القارات "Continent" وتم تقديمها من خلال الكلمة "partial" والذي يمثل الشرط اللازم للانتماء لهذه الفئة. وباستخدام البواني (instersectionOf، complementOf)، فإن القارة عبارة عن منطقة جغرافية ولكنها ليست بلد. ويتم التعبير عن أفراد هذه الفئات من خلال الكلمة "Individual". في المثال السابق تم التصريح عن أوروبا على أنها أحد الأفراد التي تنتمي إلى الفئة "قارة" التي تم تعريفها في المثال الأسبق. بمعنى آخر إن أوروبا هي قارة.

② "*Europe is a particular continent.*"	
Individual(*Europe* type(*Continent*))	*Continent*(*Europe*)

يوجد طريقة أخرى للتصريح عن فئة ما من خلال تحديد أفراد هذه الفئة بشكل مباشر كما في المثال التالي:

③ "The continents are America, Europe, Africa, Asia and Australia."

```
EnumeratedClass(Continent
    America Europe Africa Asia Australia)
```

$$Continent \equiv \{America, Europe, Africa, Asia, Australia\}$$

تم في المثال السابق تعريف الفئة "قارة" من خلال تحديد كل القيم أو الأفراد التي تنتمي إليها.

بشكل مشابه للفئات، يمكن التصريح المباشر عن الخصائص (Properties) إلى جانب الاستخدام الكيفي لبعض المعرفات الأخرى كميدان العمل (domain)، المجال (range)، وغيرها من المحددات كما في المثال التالي:

④ "Geographic regions in general contain geographic regions."

```
ObjectProperty(contains
    domain(GeographicRegion)
    range(GeographicRegion)
    inverseOf(isContainedBy)
    Transitive)
```

$$\exists\, contains.\top \sqsubseteq GeographicRegion,$$
$$\top \sqsubseteq \forall\, contains.GeographicRegion,$$
$$locatedIn \equiv isContainedBy^{-}$$
$$\mathrm{Trans}(contains)$$

تم في المثال السابق استخدام الخاصية "يحتوي (contains)" كعلاقة احتواء تعددية بين المناطق الجغرافية. وترتبط هذه الخاصية بالخاصية المعاكسة لها "محتواه ضمن (isContainedBy)". يمكن للخصائص أيضاً أن تدور في مجال يدل على نوع المعطيات كنوع معطيات الأعداد الصحيحة، الأعداد الحقيقة، أو النصية.

باستخدام الخصائص يمكن الحصول على توصيف أكثر تعقيداً للفئات من خلال تحديد بعض الشروط على هذه الخصائص كما في المثال التالي:

⑤ "A planar region only contains planar or linear regions."

```
SubClassOf(PlanarRegion
    restriction(contains
allValuesFrom(
        unionOf(PlanarRegion
LinearRegion)))
```

$$PlanarRegion \sqsubseteq \forall\, contains.(PlanarRegion \sqcup LinearRegion)$$

يظهر المثال السابق عبارة الفئة الجزئية إلى جانب بعض القيود والشروط على الخاصية التي تم تقديمها في المثال الأسبق.

يمكن التصريح عن مكافئ الفئة والذي يعتبر من العابير المعقدة لاحتوائه على قيود متداخلة على الخصائص كما في المثال التالي:

⑥ "A European city is a city whose geographic region is contained in that of Europe."

```
EquivalentClasses(EuropeanCity
intersectionOf(
    City
    restriction(locatedIn
allValuesFrom(
        restriction(isContainedBy
someValuesFrom(
            restriction isRegionFor
someValuesFrom(
            oneOf(Europe)))))))
```

$$EuropeanCity \equiv City \sqcap \forall\, locatedIn.\exists\, isContainedBy.\exists\, isRegionFor.\{Europe\}$$

يقول المثال السابق بأن المدينة تكون أوروبية إذا كانت مناطقها الجغرافية محتواه في القارة الأوروبية.

يمكن التعبير عن القيود على الخصائص من خلال وضع القيود على عددها كما في المثال التالي:

⑦ "A city is a geographic location governed by a single country."	
`SubClassOf(City restriction(governedBy maxCardinality 1))`	$City \sqsubseteq \leq 1\ governedBy$

في المثال السابق، المدن تحكم من قبل بلد واحد من خلال القيد "maxCardinality" وبشكل مشابه يمكن تحديد العدد الأصغري من خلال استخدام القيد "minCardinality".

إن أحد الاستخدامات الأخرى للخصائص هو ربط "فرد" بـ "فرد" أو بقيم كما في المثال التالي:

⑧ "Munich is a German city with 1288307 inhabitants."	
`Individual(Munich type(City) value(governedBy Germany) value(numberOfInhabitants 1288307))`	$City(Munich)$, $governedBy(Munich, Germany)$ $numberOfInhabitants(Munich, 1288307)$

تم في المثال السابق التصريح عن ميونخ على أنها مدينة تقع في ألمانيا من خلال العلاقة التي تم ربطها بها. كما وتم التصريح عن عدد صحيح يمثل عدد سكانها.

نورد في الجدول التالي مجموعة التعابير المستخدمة في الـOWL ومقابلاتها بالمنطق الوصفي.

OWL abstract syntax	DL syntax
Axioms	
`Class(A partial` $C_1 \ldots C_n$`)`	$A \sqsubseteq C_1 \sqcap \ldots C_n$
`Class(A complete` $C_1 \ldots C_n$`)`	$A \equiv C_1 \sqcap \ldots C_n$
`EnumeratedClass(A` $a_1 \ldots a_n$`)`	$A \equiv \{a_1\} \sqcup \cdots \sqcup \{a_n\}$
`SubClassOf(C D)`	$C \sqsubseteq D$
`EquivalentClasses(`$C_1 \ldots C_n$`)`	$C_1 \equiv \cdots \equiv C_n$
`DisjointClasses(`$C_1 \ldots C_n$`)`	$C_i \sqsubseteq \neg C_j, (1 \leq i < j \leq n)$
`ObjectProperty(r super(`r_1`)...super(`r_n`)`	$r \sqsubseteq r_1 \sqcap \cdots \sqcap r_n$
`domain(`C_1`) ... domain(`C_n`)`	$\exists r.\top \sqsubseteq C_1 \sqcap \cdots \sqcap C_n$
`range(`C_1`) ... range(`C_n`)`	$\top \sqsubseteq \forall r.C_1 \sqcap \cdots \sqcap \forall r.C_n$
`[inverseOf(s)]`	$r \equiv s^-$
`[Symmetric]`	$r \equiv r^-$
`[Functional]`	$\top \sqsubseteq \leq 1\, r$
`[InverseFunctional]`	$\top \sqsubseteq \leq 1\, r^-$
`[Transitive])`	$\mathsf{Trans}(r)$
`SubPropertyOf(r s)`	$r \sqsubseteq s$
`EquivalentProperties(`$r_1 \ldots r_n$`)`	$r_1 \equiv \cdots \equiv r_n$
`Individual(a type(`C_1`) ... type(`C_n`)`	$C_1 \sqcap \cdots \sqcap C_n(a)$
`value(`r_1 a_1`) ... value(`r_n a_n`))`	$r_1(a, a_1), \ldots, r_n(a, a_n)$
`SameIndividual(`$a_1 \ldots a_n$`)`	$a_1 = \cdots = a_n$
`DifferentIndividuals(`$a_1 \ldots a_n$`)`	$a_i \neq a_j, (1 \leq i < j \leq n)$

Descriptions	
Class(*A*)	A
Class(*owl:Thing*)	$\top$
Class(*owl:Nothing*)	$\bot$
intersectionOf(C_1 C_2 ...)	$C_1 \sqcap C_2$
unionOf(C_1 C_2 ...)	$C_1 \sqcup C_2$
complementOf(C)	$\neg C$
oneOf(a_1 a_2 ...)	$\{a_1\} \sqcup \{a_2\}$
restriction(r someValuesFrom(C))	$\exists r.C$
restriction(r allValuesFrom(C))	$\forall r.C$
restriction(r hasValue(a))	$\exists r.\{a\}$
restriction(r minCardinality(n))	$\geq n\, r$
restriction(r maxCardinality(n))	$\leq n\, r$

3-5. خلاصة:

قمنا من خلال هذا الفصل بالتطرق إلى تمثيل المعارف باستخدام الشبكات الدلالية والقواعد والمنطق، إلى جانب التطرق إلى ظهور الأنطولوجية في علوم الحاسب وأنواعها وأهدافها واستخدامها في الوب الدلالي الذي يتطلب بأن تكون المصطلحات الموجودة على الوب ذات معنىً واضح حتى يكون من الممكن للآلات أن تعالج هذه المعلومات أو تجهزها أوتوماتيكياً من خلال ما يتطلبه ذلك من لغات الأنطولوجية مثل الـRDF(S) والـOWL.

خدمات الوب الدلالية

Semantic Web Service

مقدّمة:

تعمل التقنيات التي يزودنا بها الوب الدلالي بهدف الوصول إلى الوب القابل للتفسير والمعالجة من قبل الآلة، ذلك الوب الذي يمكن فيه للخوارزميات البرمجية القيام بمعالجة المعلومات والاستنباط بناءً على هذه المعلومات التي كانت حكراً على التفكير البشري. من جهة أخرى، فإن تقنيات خدمات الوب تعمل بهدف إيجاد البيئة التي تقوم من خلالها المنظمات بعرض البعض من إمكانياتها وخدماتها عبر الانترنت. يتم ذلك من خلال نشر توصيف الخدمات إلى مخدم خدمات الوب (UDDI) من خلال لغة الـWSDL ليكون هذا التوصيف بمثابة العقد بين كل من مزوّد الخدمة ومستهلكها. تهدف خدمات الوب الدلالية إلى دمج هاتين التقنيتين لاستخراج الإمكانيات الكامنة في الوب من خلال إتاحة التفاعل بين نظم البرمجيات بطريقة أكثر ديناميكية.

4-1. خدمات الوب الدلالية:

تهدف خدمات الوب الدلالية إلى تسخير تقنيتي الوب الدلالي وخدمات الوب لتمكين الأنظمة من التفاعل فيما بينها بطريقة أكثر ديناميكية. تسمح خدمات الوب بتوصيف واجهة التخاطب مع الخدمة بطريقة معيارية، لكنها لا تحدد ما الذي يؤديه النظام البرمجي ولا ما هو تسلسل الرسائل المستخدم للتفاعل مع الخدمة. يمكن باستخدام تقنيات الوب الدلالي التغلب على هذا النقص. حيث يمكن إضافة وصف قابل للفهم من قبل الآلة لتلك البرمجيات التي تقدم خدماتها عبر خدمات الوب، بحيث يتم توصيف ما تفعله الخدمة وكيف تفعله. يمكن باستخدام الأنطولوجيات القادرة على توصيف الخدمات "الإعلان" عن الخدمات بطريقة غنية وقابلة للفهم من قبل الآلة مما يسمح بآلية أكثر تعقيداً لاستكشاف الخدمات مما هي عليه الآن باستخدام الـUDDI.

إن دمج تلك التقنيات سيتيح فرص جديدة. فالخدمات تتنوع من بيع كتب، ترجمة، حجوزات سفر وغيرها من الخدمات التي سيكون من الممكن الإعلان عنها واستكشافها بطريقة آلية على الانترنت. يمكن للشركة التي تبحث عن خدمة ما من إيجاد مزوّدي خدمة كامنين، الإتفاق على شروط تقديم واستلام الخدمة وطريقة الدفع مقابل الخدمة. كل هذا يمكن أن يتم بطريقة آلية وسريعة. بالإضافة إلى هذا يمكن لمجموعة من الخدمات أن تشكل خدمة جديدة مركبة أكثر تعقيداً، وربما بطريقة آلية، وفي حال عدم توافر إحدى مكوّنات هذه الخدمة المركبة فسيكون من الممكن العثور على خدمة مكافئة لها مما يحافظ على قدرة الخدمة المركبة على تأدية وظيفتها.

إن خدمات الوب الدلالي هي تقنيات تمكين، حيث أنها ضرورية لكنها وحدها غير كافية لتحقيق ما سبق. إن ماتقدمه هو توصيف الخدمات وإضافة إمكانيات لاستكشاف الخدمات والتفاعل معها. لكنها لا تقوم بالاستدلال لمعرفة ما هي الخدمة التي تريدها، من هو مزوّد الخدمة الأفضل، كيف يتم التعامل مع البارامترات وما المفروض القيام به عند استخدام الخدمة. ففي حال كانت الخدمة بسيطة وتنفيذها يتم بشكل مباشر فسيكون الاستدلال بسيطاً، ويمكن أن يكون السيناريو أكثر طموحاً. كما إن خدمات الوب الدلالية وحدها غير كافية حيث يجب استخدام تقنيات أخرى من علوم الحاسبات من أجل الوصول إلى هذا العالم الجديد.

أمثلة:

سنوضح من خلال مجموعة من الأمثلة ما يمكن أن تضيفه خدمات الوب الدلالية بالعمل إلى جانب تقنيات الاستدلال المناسبة. سيتم استخدام هذه الأمثلة للتوضيح في الفقرات القادمة أيضاً.

مثال 1: خدمة التنبيه البنكي

في هذا المثال يقدم البنك خدمة التنبيه لعملائه أثناء سحب مبالغ مالية من البنك من أجل مساعدتهم على إدارة حساباتهم حيث يقوم بإرسال إخطار للعميل من أجل تفادي السحب الزائد. ولتحقيق ذلك، يقوم نظام برمجي لدى البنك بمراقبة حساب العميل وطريقة التعامل معه، متى يتم الإيداع في هذا الحساب ومتى يتم السحب منه وبأي تردد وبأية كميات. ثم وبالاعتماد على الرصيد الحالي والمناقلات المستقبلية المتوقعة سيكون من الممكن لهذا النظام تحديد فيما إذا سيصل الحساب إلى مرحلة السحب الزائد، عندها سيتم تحذير العميل عن طريق إرسال رسالة نصية أو صوتية مباشرة. لإرسال هذه الرسالة يقوم النظام البرمجي لدى البنك باستخدام خدمة ما لإرسال الرسائل وبدون علم مسبق بمزوّد هذه الخدمة، وإنما يتم بشكل أتوماتيكي أثناء التشغيل يتم اختيار الخدمة ومزوّدها من دليل الخدمات المتاحة بناء على عوامل

اختيار ما مثل التكلفة وخيارات إرسال الرسائل. عندها يقوم النظام بإرسال فحوى هذه الرسالة إلى مزوّد هذه الخمة والذي يقوم بدوره بإرسالها إلى العميل مباشرة.

مثال 2: الشراء الذكي

تقوم شركة تصنيع كبيرة بعمليات شراء منتظمة للمواد التي تحتاجها من مجموعة متعددة من الموردين لديهم تواجد على الشبكة العنكبوتية. يتم شراء المواد الأساسية وذات الأهمية الأكبر عبر سلسلة موردين محددين ومقبولين لدى الشركة ولديهم التزامات معينة بتلبية احتياجات الشركة بالوقت المناسب. أما بالنسبة للمواد الأقل أهمية وخطورة على استمرار الأعمال كالقرطاسية مثلاً فيمكن شراؤها من قبل أي مورد يتمتع بالسمعة الجيدة مما يتيح إمكانية الشراء بأفضل قيمة.

يتم تزويد وكيل برمجي ينوب عن الشركة بقائمة من مستلزمات القرطاسية للشهر القادم، يقوم هذا الوكيل بالبحث ضمن دليل الموردين المقبولين من قبل الشركة لتزويد القرطاسية. يقوم الموردون بتأمين خدمات الشراء من خلال مواقعهم الالكترونية حيث يمكن للزبون استعراض المواد المتاحة واختيار المواد التي يريد شراؤها ثم الدفع بالطريقة المحددة. بنفس الوقت يقدم الموردون خدمات الشراء هذه بشكل خدمات وب تسمح للبرامج الأخرى أو الوكلاء بالتعامل معها واستخدام هذه الخدمات بنفس الطريقة المقدّمة للزبائن. حيث يقوم الوكيل البرمجي بزيارة هذه المواقع والبحث عن المواد المفوَّض بشرائها ومدى توافرها لدى الموردين ثم يقوم بإعداد مقترح طلبات شراء لدى عدة موردين مع الحسم إن وجد، ثم يقوم باختيار المورد المناسب بناءً على العروض المقدمة ويتقدم بطلب الشراء كما يقوم بإلغاء الطلبات الأخرى. عندها يقوم المورد بإرسال الطلبية للشركة.

مثال 3: سلسلة التوريد اللوجيستي

تحتاج شركة إلى إجراء عملية شحن لبضاعة من دمشق إلى الرباط. تملك هذه الشركة عقد طويل الأمد لنقل هذا النوع من البضائع مع مورد ما من دمشق إلى القاهرة وكذلك من تونس إلى الرباط. لسبب ما فإن المورد المعتاد لنقل البضاعة بين القاهرة وتونس غير قادر على تلبية هذه الحاجة في هذا الوقت، بالتالي هناك حاجة ملحّة لإيجاد البديل المناسب وبالسرعة المناسبة. يتم تزويد وكيل برمجي ينوب عن الشركة بمعلومات عن مهمة الشحن هذه. يقوم هذا الوكيل بالاتصال بوكيل استكشاف ما يملك القدرة على الوصول إلى توصيف خدمات الشحن التي يقدمها مزوّدي خدمة ما، يقوم مخدم الاستكشاف هذا بسؤال مزوّدي الخدمات عن إمكانية تقديم خدمة للشحن بين القاهرة وتونس ويعود بقائمة من المزوّدين المحتملين إلى الوكيل البرمجي. يقوم الوكيل البرمجي باختيار مزوّد خدمة محتمل أو أكثر ويقوم بإرسال تفاصيل مهمة الشحن التي يحتاجها متضمنةً تاريخ وصول الشحنة إلى القاهرة وتاريخ الوصول المفترض إلى تونس. يقوم مزوّد الخدمة بالإجابة عن إمكانية الاستجابة لهذه المتطلبات فمثلاً: يمكن لمزوّد الخدمة تقديم عرض بتوصيل الشحنة لكنها ستصل قبل يوم من التاريخ المحدد من قبل الشركة مع التكلفة.

يقوم الوكيل البرمجي باختيار أحد الخدمات المقترحة (ربما بالتعاون مع المستخدم للوصول إلى القرار النهائي) ويقوم بإخطار مزوّد الخدمة بقراره مما يعني وصولهم إلى إتفاق للشحن. وعندما يتم الشحن المطلوب، يتم تبادل رسائل معيارية بين الطرفين للتأكيد على الاستلام والتسليم.

مثال 4: أسعار المؤشرات

يهتم أحد المستثمرين بالبقاء على إطلاع دائم بأسعار الأسهم. يمكنه ذلك من خلال تقنيات خدمات الوب، عندما يتصل بالانترنت يقوم برنامج بالبحث عن الخدمات التي تزود المستخدمين بآخر أسعار الأسهم ويتم إيجاد خدمتين ويطلب من المستخدم إختيار أحدها، حيث تقدم الخدمة الأولى آخر الأسعار لكن بتأخير مقداره دقيقة مقابل 1000 ليرة سورية في الشهر، الخدمة الأخرى مجانية لكن بتأخير 30 دقيقة. ولأن المستثمر غير مهتم بالحصول على المعلومة بالزمن الحقيقي(real time) فسيقوم باختيار الخدمة الثانية المجانية، حينئذٍ يقوم البرنامج بتحديث معلومات المستثمر كلما اتصل بالانترنت.

4-1-1. مفاهيم أساسية في خدمات الوب الدلالية:

سنعرض بعض المفاهيم الأساسية لخدمات الوب الدلالية وعلاقاتها ببعضها:

- **الخدمة (Service):**

تقوم جهة ما بتزويد خدمة لجهة أخرى عندما تتولى القيام بعمل ما لصالح هذه الجهة ويمكن أن تكون هذه الخدمة مجانية وغالباً ما تكون مدفوعة الأجر. فمنظف الزجاج يقوم بخدمة تنظيف الزجاج، الحلاق يقوم بخدمة قص الشعر. وفي المثال الأول السابق يقدم البنك خدمة التنبيه البنكي لعملائه، وفي المثال الثاني يقدم المورد خدمة بيع القرطاسية وشحنها إلى الشركة، وفي المثال الثالث يقدم مزوّد الخدمة خدمة شحن البضاعة بين مينائين.

يمكن القول بأن الخدمة هي قيام طرف ما ببعض الأعمال التي تعني قيمة ما لصالح طرف أخر. ويتم عادةً الحديث عن الخدمة ضمن مجال ما ففي المثال الثالث السابق يتم تقديم الخدمة ضمن مجال النقل والدعم اللوجيستي، ويشار لذلك بمجال القيمة التي تضيفها الخدمة (*domain of value*).

نطلق على الجهة التي تقدم الخدمة بـ مزوّد الخدمة والجهة المستفيدة من الخدمة بـ طالب الخدمة أو مستهلك الخدمة. ويمكن النظر للخدمة من خلال عدة مستويات من التجريد:

فالخدمة المحددة (*concrete service*) وهي تنفيذ أعمال محددة في وقت محدد من قبل جهة ما لصالح جهة أخرى. مثلاً في المثال الثالث السابق تكون الخدمة المحددة هي القيام بشحن البضاعة في الحاوية رقم 1132 من القاهرة بتاريخ 01/08/2011 الساعة 9:00 إلى تونس بتاريخ 03/08/2011 الساعة 10:15.

أما الخدمة المجردة (*abstract service*) فتتعلق بمجموعة أو فئة معينة من الخدمات المحددة وتسمح لنا بمناقشة الخدمات المستقبلية المفترضة بدون تقديم تفاصيل دقيقة عن كل شيء، ففي المثال الثالث السابق يرغب طالب الخدمة بالحديث عن خدمة مفترضة يتم فيها شحن الحاوية ذات الرقم 197 في وقت ما من القاهرة يوم 01/08/2011 إلى تونس قبل تاريخ 05/08/2011.

- **تمثيل الخدمة (Service Representation):**

إن أحد أهداف خدمات الوب الدلالية هو تمثيل الخدمات بشكل قابل للقراءة والفهم من قبل الآلة وفقاً للقيمة التي تضيفها هذه الخدمة. يشار إلى ذلك بتوصيف الخدمة(*service description*). عادةً ما يقدم توصيف الخدمة وصفاً مجرداً للخدمة يمكن أن يشار إليه بالتوصيف المجرد للخدمة. لتوصيف الخدمة، يستخدم الوب الدلالي تقنيات تمثيل المعارف، والتي توفر مجموعة من اللغات الصورية والتقنيات لتوصيف المعارف بطريقة تسمح بالاستدلال في مجموعة المعارف هذه. عند توصيف

الخدمة يجب تحديد أمرين: الأول إختيار اللغة الصورية التي سيتم من خلالها التوصيف والثاني هو تحديد مجموعة المفاهيم والعلاقات التي سيتم توصيفها. ويتطلب هذا خلق واختيار الأنطولوجية، والتي توفر مجموعة مصطلحات أنطولوجية بشكل مهيكل: مجموعة من المفاهيم والعلاقات المستخدمة لتوصيف الأشياء في مجال العمل المستهدف.

عندما تقوم جهتين مختلفتين بتوصيف خدمة ما فستقوم كل منهما باختيار اللغة والأنطولوجية المختلفة عن اختيار الآخر. كنتيجة لذلك إذا أراد أحد هاذين الطرفين الاستدلال من خلال التوصيف المزوّد من قبل الطرف الأخر فسنكون هنا بحاجة إلى استدلال إضافي للقيام بالترجمة بين طريقتي التوصيف المعتمدتين من قبل هاذين المزوّدين. يطلق على هذا الاستدلال الإضافي بالوساطة (*mediation*)، وخاصة الوساطة في الأنطولوجية (ontology mediation). يمكن أن نلاحظ وجود أشكال أخرى للوساطة يمكن ذكرها لاحقاً عند الحديث عنها.

- **الوكلاء (Agents):**

تتم الإشارة إلى المكوّنات البرمجية التي تمثل كل من مزوّد الخدمة وطالبها وتنوب عنهما بالوكلاء. تتصرف الوكلاء كممثلين عن الجهة التي يمثّلونها على الشبكة. يتم وصف الوكلاء وفقاً للدور الذي يلعبونه، لذلك في حالتنا فالوكلاء هنا ليسو كالوكلاء المتعارف عليهم في النظم متعددة الوكلاء[Heard 2005،Beale]، حيث أنهم غالباً سيكون منفعلين وليسو فاعلين وسيتبعون خطوات عمل محددة بشكل مسبق. فعلى سبيل المثال إن مجموعة خدمات الوب التي تقدمها شركة أمازون ينوب عنها وكيل يقوم ببيع الكتب، يمكن أن يكون هذا الوكيل أكثر تطوراً باستخدام تقنيات النظم متعددة الوكلاء ويقوم بعملية التفاوض باستخدام خوارزميات التفاوض [Heard،Beale ، Lloyd 2005،Kalra] مع الوكلاء الأخرين حول الخدمة وسعرها، كما ويمكن أن يقوم بعملية التخطيط الموزّع [Kabak 2005، Dogac، Laleci،Bicer] للسماح للخدمات المركبة المعقدة، كما ويمكن استخدام نظريات الإستدلال لاختيار أحد البدائل المتاحة لعمل ما أثناء توصيل الخدمة [Wilken 2005،Boniface].

وكنتيجة لكون الوكيل هو ذلك المكوّن البرمجي فيمكنه أن يتصرف كطالب خدمة ويقوم بتزويد خدمة في نفس الوقت. ففي المثال الأول السابق يتصرف النظام البرمجي للبنك كمزوّد خدمة لعملائه مقدماً رسائل تحذير تتعلق بحساباتهم، وللقيام بذلك يتصرف هذا النظام وفي نفس الوقت كطالب لخدمة التحذير من قبل مزوّد خدمة آخر.

- **الاتصال (Communication):**

عندما يتعامل مزوّد الخدمة وطالب الخدمة مع بعضهما فهما بحاجة إلى تبادل مجموعة من الرسائل. تبادل الرسائل هذه يجب أن يكون وفقاً لقواعد معينة أي وفقاً لبروتوكول اتصال متعارف عليه بين الطرفين. يشار إلى بروتوكول الاتصال هذا في عالم خدمات الوب الدلالية بالبروتوكول *"choreography"*. يشار إلى عملية تبادل الرسائل بين الطرفين وفقاً للقيود المحددة ضمن البروتوكول بالمحادثة (conversation) بين هذه الأطراف. ويمكن أن تتفاعل مجموعة من الأطراف مع بعضها البعض مما يعقّد عملية الإتصال أكثر.

كما ذكرنا سابقاً بأن الهدف التقني الأول لخدمات الوب الدلالية هو تأمين توصيف للخدمات قابل للفهم من قبل الآلة، للسماح للإستدلال من خلالها من قبل أطراف مختلفة. أما الهدف التقني الثاني لخدمات الوب الدلالية فهو توصيف البروتوكولات المختلفة والتي يتم استخدامها من قبل هذه الأطراف للتعامل

مع بعضها البعض وبطريقة مفهومة من قبل الآلة أيضاً. إن هذا لا يشمل فقط الرسائل التي يجري تبادلها وإنما أيضاً النموذج أو الطريقة التي يجري فيها تبادل تلك الرسائل بين مختلف الأطراف، بمعنى أخر يجب أن تمثل المعنى الدلالي لتبادل الرسائل. ففي المثال الثاني السابق يتم تبادل مجموعة من الرسائل من أجل بناء أو تشكيل طلبية القرطاسية، فالتمثيل الدلالي يمكن أن يظهر أن استخدام سلسلة معينة من الرسائل يعني إضافة مادة جديدة إلى الطلبية، بينما تسلسل أخر من الرسائل يعني الحصول على العرض أو المناقصة، وتسلسل أخر يعني الوصول إلى إتفاق نهائي لمعالجة طلب الشراء والدفع.

إن القيام بذلك سيمكّن هذه البرمجيات من الإستدلال عبر البروتوكول "choreography". فعلى سبيل المثال سيكون من الممكن تحديد ما هي الخطوة التالية التي يجب على البرنامج إتخاذها وما هي الرسالة التالية الواجب إرسالها.

- **الوساطة (Mediation):**

عندما يتم التخاطب والتفاعل بين أكثر من طرف لابد من وجود وسيط يوفّق فيما بينهم. هناك أربعة أنواع من الوسطاء: على مستوى البيانات، على مستوى الأنطولوجية، على مستوى البروتوكول "choreography"، وعلى مستوى العملية (Process).

1- على مستوى البيانات (Data Mediation):
تقوم الرسالة أو جزء البيانات بتمثيل المعلومات التي تحملها بصيغة نحوية محددة. لذلك فيمكن لمزوّدي خدمة مختلفين إستخدام صيغ نحوية مختلفة للتعبير عن الرسائل التي تستخدمها الخدمات التي يقدمونها، حتى وإن كانت تحمل نفس المعلومات. يهتم وسطاء البيانات بالتحويل بين هذه الصيغ النحوية المختلفة.

2- على مستوى الأنطولوجية (Ontology Mediation):
عندما يقوم طرفان بتوصيف خدماتهما قد يختاران مصطلحات مختلفة وبالتالي أنطولوجية مختلفة. فإذا أراد أحد الأطراف الإستدلال في التوصيف المزوّد من قبل الطرف الأخر ستظهر الحاجة إلى وجود طبقة ثانية من الإستدلال لتقوم بالترجمة بين هاتين المقاربتين المختارتين من قبل هذين الطرفين. يطلق على هذا الإستدلال الثاني مصطلح الوسيط الأنطولوجية.

3- على مستوى البروتوكول (Protocol Mediation):
بسبب الاختلاف الذي غالباً ما يتواجد في تصميم بروتوكول تسلسل الرسائل التي من الممكن تبادلها بين أطراف الخدمة (المزوّد والمستهلك) حتى وفي حال اتباع مصممي البروتوكول لدى كلٍ من الطرفين لمعيار معين فلا بد من وجود اختلاف بين التصميمين، بسبب هذا فلا بد من وجود وسيط هو وسيط البروتوكول والذي يقوم بدور التسوية بين البروتوكولات من خلال قيامه بترجمة تسلسل الرسائل المستخدمة من قبل أحد الأطراف إلى تسلسل رسائل مختلف مستخدم من قبل الطرف الأخر.

4- على مستوى العملية (Process Mediation):
يملك كل طرف من أطراف الخدمة عملياته الداخلية التي تدير عملية الإستدلال والموارد اللازمة لإنجاح عملية التخاطب مع الطرف الآخر. حتى وإن كان الطرفان قادرين على التخاطب بواسطة بروتوكول ما ستبقى هناك فجوة بسبب الإختلاف في عمليات كل طرف عن العمليات الموجودة في الطرف الآخر، مما يفشل عملية التخاطب بين هذين الطرفين. إن وسيط العمليات يقوم بالتسوية بين العمليات المختلفة ويعدّ أصعب أنواع الوساطة وفي الكثير من الحالات قد لا ينجح ويتطلب إعادة هندسة العمليات (Process re-engineering).

إن الأنواع الأربعة من الوسطاء لا يمكن أن تتم بشكل أتوماتيكي إلا إذا تم تمثيل البروتوكول والرسائل بشكل دلالي مما يسمح بإمكانية التفاعل والتخاطب بين أطراف الخدمات بشكل أتوماتيكي.

- **دورة الحياة (Life Cycle):**

تمر العلاقة بين كل من مزوّد الخدمة وطالب الخدمة بشكل عام عبر أربع مراحل: النمذجة، الاستكشاف، تعريف الخدمة، توصيل الخدمة. خلال الاستكشاف يحاول طالب الخدمة تحديد المزوّدين المحتملين القادرين على الاستجابة لمتطلباته. خلال تعريف الخدمة يتخاطب كل من المزوّد والطالب للخدمة مع بعضهما لوضع تفاصيل الخدمة المطلوب تأمينها. خلال توصيل أو تسليم الخدمة تجري مجموعة من التفاعلات بين الطرفين في سبيل تأمين المطلوب من الخدمة.

1- مرحلة نمذجة الخدمة (Service Modeling Phase):
قبل البدء بعملية استكشاف الخدمات، على طالب الخدمة أن يقوم بتحضير توصيف للخدمة المهتم بها ويريد البحث عنها. غالباً لن يحتوي هذا التوصيف على كامل التفاصيل الدقيقة فمزوّد الخدمة غير معروف وسعر الخدمة غير معروف أيضاً، لذلك سيكون توصيف لخدمة مجردة. إن هذا التوصيف المجرد للخدمة يمثّل توصيف متطلبات الخدمة التي يطلبها طالب الخدمة ويبحث عنها. بشكل مشابه، يقوم مزوّد الخدمة بخلق توصيف مجرد للخدمة التي يقدمها. يعبر عن هذا التوصيف بتوصيف عرض الخدمة. من الممكن ملاحظة أن توصيف متطلبات الخدمة وتوصيف عرض الخدمة يشكلان في النهاية توصيف للخدمة وبالتالي يستخدمان نفس المفاهيم والعلاقات، لكن لكل منهما وظيفة مختلفة عن الآخر، فالأول يوصف الخدمة التي يجري البحث عنها والآخر يمثّل الخدمة التي يتم تزويدها.

2- مرحلة استكشاف الخدمة (Service Discovery Phase):
عند توافق كل من توصيف متطلبات الخدمة المقدم من قبل طالب الخدمة وتوصيف عرض الخدمة المقدم من قبل مزوّد الخدمة عندها نقول بأن هناك حالة تطابق ويمكن للطرفين البدء بمرحلة تعريف الخدمة فيما بينهما. يوجد عدة طرق لتحديد وجود التوافق بين توصيفين مختلفين للخدمة. ففي المثال الأول السابق، يبحث البنك عن مزوّد خدمة لإرسال رسالة إلى عميل، ولنفترض أن العميل اختار أن يكون شكل الرسالة نصي. إن وكيل البنك الطالب للخدمة سيقوم بخلق توصيف متطلبات الخدمة يورد فيها الحاجة إلى إرسال رسالة نصية بطول 110 أحرف إلى رقم في شبكة خلوي معينة في سوريا. في الوقت الذي يعلن فيه مزوّد الخدمة عن توصيف عرض خدمة يورد فيه قدرته على إرسال رسائل نصية بطول أعظمي مقداره 120 حرف إلى أية شركة خلوي في سوريا وبتكلفة 30 ليرة سورية للدقيقة. من الواضح أن هذين الوصفين متوافقين مما يعني إمكانية التعامل بين طالب الخدمة ومزوّدها.

يوجد العديد من البنى التي يمكن أن يتم الإستكشاف من خلالها. الطريقة الأكثر شيوعاً هو وجود خدمة استكشاف مركزية يتعامل معها طالب الخدمة من خلال بروتوكول تبادل رسائل بسيط.

3- مرحلة تعريف الخدمة (Service Definition Phase):
من الممكن في مرحلة الإستكشاف أن يحدد طالب الخدمة مجموعة من مزوّدي الخدمة المحتملين لتلبية احتياجاته. يقوم طالب الخدمة بالإتصال بواحد أو أكثر من هؤلاء المزوّدين والانتقال إلى تعريف الخدمة المطلوبة. يمكن أن تكون عملية إختيار المزوّد هذه عشوائية بكل بساطة، أو قد تتبع لقواعد معينة يحددها طالب الخدمة، ففي المثال الثالث السابق قام طالب الخدمة بتفضيل اختيار الخدمة الأرخص بالرغم من أن معلوماتها أقدم، ولكن كان لسعر الخدمة الإهتمام الأكبر بالنسبة له على حساب مدى حداثة المعلومات. في حال فشلت عملية تعريف الخدمة مع مزوّد الخدمة الذي تم اختياره فيمكن

لطالب الخدمة اختيار مزوّد أخر من القائمة الأساسية لمزوّدي الخدمة المحتملين ويعيد المحاولة من جديد.

يتضمن تعريف الخدمة إعادة صياغة للتوصيف المجرد للخدمة لإضافة نفاصيل محددة أكثر عن الخدمة بما يلبي إحتياجات طالب الخدمة. ففي المثال الثالث السابق فإن خدمة الشحن تتطلب مجموعة من المواصفات مثل وزن الحاوية المراد شحنها، موانئ المغادرة والوصول، أوقات المغادرة والوصول، والأسعار. إن تحديد قيم هذه المواصفات يتم في مرحلة تعريف الخدمة. تتم هذه الخطوات من خلال المحادثة بين الطرفين والتي تكون مضبوطة من خلال بروتوكول تعريف الخدمة.

قد يقوم طالب الخدمة في مرحلة تعريف الخدمة بالمحادثات مع أكثر من مزوّد خدمة بنفس الوقت وذلك بهدف الإطلاع على الخيارات المتاحة أمامه وتحصيل أفضل عرض ممكن. ففي المثال الثاني السابق فإن طالب خدمة شراء القرطاسية يقوم بنحضير طلبات شراء من مزوّدين متعددين ثم يقوم باختيار العرض الأفضل ويلغي العروض أو الطلبات الأخرى غير الناجحة. وفي حال نجاح إتمام مرحلة تعريف الخدمة بين الطرفين يمكن الانتقال إلى مرحلة تسليم الخدمة.

تجدر الإشارة إلى أنه من الممكن أن لا يتم في مرحلة تعريف الخدمة تحديد قيم كل المواصفات وإنما تقييدها بشرط ما فقط، ففي المثال الثالث السابق فإن مزوّد الخدمة يقوم بتحديد الحد الأعظمي لوزن الحاوية المطلوب شحنها بـ600 كغ، مما يعني بأن مزوّد الخدمة يسمح هنا بإمكانية تأجيل تحديد قيمة مواصفة الوزن إلى مرحلة التسليم.

في العديد من الحالات قد يكون من غير الضروري المرور بتعريف الخدمة، حيث يقوم مزوّد الخدمة بتحديد كل القيم التي تهمه في توصيف الخدمة التي يقدمها. إن التوصيف الأكثر مرونةً هو الذي يتيح لطالب الخدمة الإختيار بحرية. ففي المثال الرابع السابق لا يوجد محادثات لتعريف الخدمة بين برنامج المستثمر وبين مزوّد خدمة أسعار الأسهم، حيث يقوم المستثمر بالإختيار وفقاً لخياراته بكل بساطة. في حالات أخرى يتم الدخول في مرحلة تعريف الخدمة بشكل تكراري، أي الاختيار من بين الخيارات المتاحة جزء بجزء. ففي المثال الثاني السابق ومن خلال تبادل الرسائل بين طرفي الخدمة يمكن لطالب الخدمة إستعراض السلع، إختيار بعضها، طلب العرض النهائي والاتفاق على شراء السلع أو لا.

يمكن أحياناً أن تتخلل عملية المحادثة بين الطرفين عملية تفاوض على بارامترات معينة كالسعر مثلاً.

4- مرحلة تسليم الخدمة (Service Delivery Phase):

يمكن عند الإتفاق على تعريف الخدمة بين الطرفين الإنتقال إلى مرحلة تسليم الخدمة. يمكن أن يتم التسليم بشكل فوري كما في المثال الأول السابق حيث يتم إرسال رسالة التحذير بشكل مباشر. وقد تتم عملية التسليم بعد فترة من الإتفاق على تعريف الخدمة، ففي المثال الثالث السابق يمكن أن تتم بعد الإتفاق بفترة جيدة. ويمكن أن يتم تسليم الخدمة بدون أي تواصل بين الطرفين ففي المثال الأول السابق يتم إرسال رسالة التحذير مباشرةً وبدون أي تبادل للرسائل بين الطرفين بعد الإرسال. وأيضاً من الممكن أن يكون هناك نوع من الإتصال بين الطرفين والذي يتم ضبطه من خلال بروتوكول التواصل.

يوجد عدة أنواع من التواصل أو التفاعل بين طرفي الخدمة يمكن أن تتم خلال مرحلة تسليم الخدمة، ويخضع كل منها لبروتوكول:

- يغطي بروتوكول تسليم الخدمة الرسائل التي يتم تبادلها أثناء تسليم الخدمة مباشرةً. ففي المثال الرابع السابق يتم إرسال معلومات الأسهم ضمن رسالة الرد التي ترسلها خدمة الوب بشكل مباشر.

في حالات أخرى يرتبط تبادل الرسائل بأحداث تجري على أرض الواقع ففي المثال الثالث السابق يتم تحديد الرسائل المتبادلة بحركة البضائع التي يجري نقلها بين القاهرة وتونس.

- يغطي بروتوكول المراقبة الرسائل التي يتم تبادلها والذي يسمح لطالب الخدمة باستلام معلومات عن حالة الخدمة التي يجري تسليمها من قبل مزوّد الخدمة. ففي المثال الثالث السابق يستلم طالب الخدمة رسائل تحوي معلومات عن حالة الشحن الجارية.
- يسمح بروتوكول الإلغاء أو إعادة التفاوض لطالب الخدمة في بعض الحالات بإلغاء أو تعديل الخدمة التي يستلمها من مزوّد هذه الخدمة. فيمكن مثلاً إلغاء عملية طلب شراء لكتاب إذا لم تبدأ عملية إرساله لطالب الخدمة.

2-1-4. بنى خدمات الوب الدلالية:

سيتم التطرق إلى البنية التي يمكن استخدامها لتطوير وتطبيق التطبيقات المعتمدة على خدمات الوب الدلالية. يتم في أبحاث النظم متعددة الوكلاء التمييز بين بنى الـMicro والـMacro. وتمثل البنية الأولى بنية داخلية تعتمد على المكوّنات ضمن المجتمع نفسه. بينما تمثل البنية الثانية الهيكل الكلي للمجتمع مع إعتبار كل وحدة فيه كصندوق أسود. سيكون من المفيد إعتماد نفس البنى في خدمات الوب الدلالية بالشكل التالي:

1- بنية الـMacro (Macro Architecture):

يمكن للوحدة البرمجية في مجتمع خدمات الوب الدلالية لعب أحد ثلاثة أدوار وهي: وكيل طالب للخدمة، وكيل مزوّد للخدمة، ووكيل مستكشف الخدمة. بشكل عام يمكن للوحدة لعب أكثر من دور في نفس الوقت لكننا للوضوح سنتكلم عن الأدوار منفصلةً.

يتصرف وكيل طالب الخدمة بالنيابة عن أفراد أو منظمات لطلب خدمة. يتسلم توصيف الخدمة من الجهة الموكّلة له ويقوم بالتخاطب مع وكلاء آخرين محاولاً تلبية تلك المتطلبات. يمتلك هذه الوكيل نموذج ما خاص بمجال عمل الخدمة وهو الأنطولوجية بالإضافة إلى نماذج أخرى عن أنواع الأفعال الممكنة في هذا المجال والممثّلة بتبادل الرسائل.

وكيل مزوّد الخدمة ينوب عن المنظمة بتقديم الخدمة. يمتلك هذا الوكيل توصيف عرض الخدمة الموصّف باستخدام الأنطولوجية "في الحالة المثالية تكون هذه الأنطولوجية مشابهة للانطولوجي المستخدمة من قبل طالب الخدمة"، وتوصيف الخدمة القادرة على تزويدها لطالبي الخدمة بمستوىً من التجريد. كما ويمتلك بالإضافة لذلك توصيف بروتوكول تسليم الخدمة.

أما وكيل استكشاف الخدمات فيمكنه الوصول إلى توصيف عروض الخدمات مع معلومات عن الوكيل المزوّد لهذه الخدمات.

يمكن توضيح بنية الـMacro من خلال توصيف التفاعل الحاصل بين هذه الأنواع المختلفة من الوكلاء بالشكل التالي:

- الوكيل المزوّد للخدمة يقوم بتسجيل توصيف عرض الخدمة:

يتم استخدام بروتوكول تبادل رسائل بسيط بين الوكيل المزوّد والوكيل المستكشف. يرسل الوكيل المزوّد رسالة تسجيل خدمة إلى الوكيل المستكشف محتويةً على توصيف عرض الخدمة بالأنطولوجية

الخاصة بالوكيل المستكشف بالإضافة إلى عنوان الوكيل المزوّد للخدمة "URI". يقوم الوكيل المستكشف بإرسال رسالة إجابة بقبول التوصيف وتخزينه لديه أو برفضه. يتم الرفض إذا كان التوصيف غير صحيح بالنسبة لأنطولوجية المستكشف أو يتم الرفض لأي سبب أخر لعدم توافر الذاكرة لدى المستكشف على سبيل المثال.

في حال جهل مزوّد الخدمة بالأنطولوجية الخاصة بالمستكشف فيقوم عندها بإرسال رسالة طلب الأنطولوجية "requestOntology" إلى المستكشف الذي يقوم بدوره بالرد برسالة "informOntology" والتي تحتوي على أنطولوجية توصيف الخدمات. إذا كانت هذه الأنطولوجية مختلفة عن الأنطولوجية المستخدمة من قبل مزوّد الخدمة فتظهر الحاجة بينهما إلى وجود وسيط الأنطولوجية (Ontology Mediation) الذي تطرقنا له سابقاً. يمكن أن تتم عملية الوساطة هذه لدى مزوّد الخدمة أو المستكشف أو حتى لدى طرف ثالث (third party).

- الوكيل الطالب للخدمة يبحث عن المزوّدين المحتملين:

يتم البحث عن الخدمة من خلال بروتوكول تبادل رسائل بسيط بين طالب الخدمة ومستكشف الخدمة. يقوم الوكيل الطالب للخدمة بإرسال رسالة "طلب مزوّدي خدمة requestProviders" تحتوي على توصيف متطلبات الخدمة باستخدام أنطولوجية مستكشف الخدمات إلى وكيل استكشاف الخدمات (وكما في الخطوة السابقة يمكن لطالب الخدمة استخدام رسائل requestOntology وinformOntology للتعرف على أنطولوجية المستكشف وقد تظهر الحاجة إلى استخدام وسيط الأنطولوجيات في حال وجود اختلاف بين الأنطولوجية المستخدمة من قبل طالب الخدمة ومستكشف الخدمة)، ويقوم المستكشف بإرسال رسالة الإجابة "informProviders" محتوية على قائم بالمزوّدين المحتملين للخدمة مع عناوينهم "URIs". أولئك المزوّدين لديهم عروض خدمات مطابقة لمتطلبات الخدمة المطلوبة باستخدام خوارزميات المطابقة.

- مزوّد وطالب الخدمة يتفقان على تعريف الخدمة:

بعد مرحلة استكشاف الخدمات، يقوم وكيل طالب الخدمة بتبادل مجموعة من الرسائل مع أحد المزوّدين المحتملين للخدمة أو أكثر بهدف تعريف الخدمة التي سيتم تسليمها ثم لاختيار المزوّد المناسب. في البنية التي نتكلم عنها سيتم للتبسيط إفتراض أن كل وكلاء طالبي ومزوّدي الخدمات سيقومون باستخدام بروتوكول بسيط واحد لتعريف الخدمة فيما بينهم (بشكل عام إن هذا الإفتراض قد يكون غير واقعي، بل وقد تكون هناك حاجة لاستخدام وسيط البروتوكولات أيضاً). يمكن استخدام معايير الـFIPA [Valle، Cerizza 2005] والتي تزود مجموعة من بروتوكولات التفاوض التي يمكن استخدامها في هذه المرحلة. يتكون البروتوكول البسيط بالشكل التالي:

يقوم الوكيل الطالب للخدمة بإرسال رسالة طلب خدمة "requestService" إلى كل مزوّد خدمة وتحتوي هذه الرسالة على توصيف المتطلبات. يقوم مزوّد الخدمة بالإجابة برسالة "informService" والتي تحتوي على توصيف الخدمة أو الخدمات القادر على تزويدها. إذا رغب طالب الخدمة باستخدام أحد هذه الخدمات يقوم بإرسال رسالة اختيار خدمة "selectService" وتحتوي على الخدمة التي تم اختيارها ويقوم المزوّد بالإجابة برسالة تأكيد "confirmService" وتحتوي على عنوان توصيف بروتوكول استخدام الخدمة الذي سيتم استخدامه عند تسليم الخدمة. إذا لم يُجب طالب الخدمة بطلب استخدام أحد الخدمات خلال وقت محدد فيتم إلغاء طلبه. نلاحظ بعدم وجود إمكانية التفاوض في هذا البروتوكول البسيط وإنما يسمح لمزوّد الخدمة بإعداد قائم بالخدمات التي يمكن لطالب الخدمة اختيارها ليقوم طالب الخدمة باختيار أحد هذه الخدمات.

- تسليم الخدمة:

يبدأ تسليم الخدمة عندما يقوم أحد الطرفين (وفقا لبروتوكول التسليم) بإرسال رسالة بدء التسليم إلى الطرف الآخر. بخلاف المرحلة السابقة يمكن هنا استخدام العديد من البروتوكولات وفقاً لمجال العمل الذي يتم خلاله تقديم الخدمة. ففي المثالين الأول والرابع الذي عرضا سابقاً، البروتوكول بسيط ويتألف من رسالة واحد تفيد بطلب تنفيذ الخدمة مع رسالة إجابة بالانتهاء من تنفيذ الخدمة. أما في المثال الثالث فيتكون البروتوكول من تبادل عدة رسائل بين الطرفين.

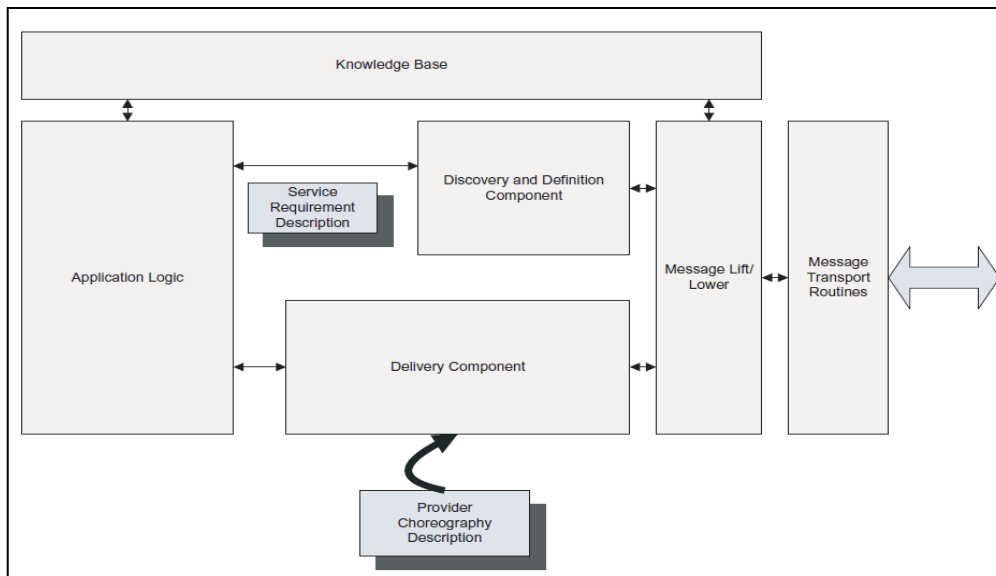
بسبب التنوع الكبير في بروتوكولات مرحلة التسليم ستكون هناك حاجة لاستخدام وسيط البروتوكولات (Protocol Mediation) وخاصة عندما يكون البروتوكول معقّداً كما هي الحال في المثال الثالث السابق سنفترض في هذه البنية بأن الوساطة ستتم لدى طالب الخدمة ووفقاً للتوصيف المقدم من قبل مزوّد الخدمة علماً بأن ذلك يمكن أن يتم من قبل طرف ثالث.

بناءً على هذا الافتراض فإن بنية الـMacro ستكون بالشكل التالي:

- كل مزوّد خدمة لديه توصيف لبروتوكول تسليم الخدمة من أجل كل خدمة قادر على تزويدها.
- في نهاية بروتوكول تعريف الخدمة يقوم مزوّد الخدمة بتزويد عنوان "URI" بروتوكول تسليم الخدمة هذا.
- يكون طالب الخدمة مسؤول عن الوصول إلى هذا التوصيف وتفسيره ويقوم بتبادل الرسائل مع المزوّد بما يتوافق مع متطلبات هذا التوصيف.

2- بنية الـMicro (Micro Architecture):

سنطّلع على هذه البنية لدى كل من وكيل طالب الخدمة ووكيل مزوّد الخدمة. على العكس من بنية الـMacro فإن بنية الـMicro ليست على مستوى المجتمع.حيث يتم في البنية الأولى تعريف تبادل الرسائل بين أطراف يلعبون أدوار مختلفة، ولكي يعمل المجتمع بالطريقة الصحيحة يجب الاتفاق والالتزام بهذه الرسائل من قبل جميع الأطراف وقد يتطلب ذلك تزويد وسطاء على مستوى البروتوكول لمنح المزيد من المرونة. أما بنية الـMicro لكل وكيل فلا تحتاج إلى مثل هذا الاتفاق المسبق على البنية. يوضح الشكل "1-4" البنية الداخلية لوكيل طالب الخدمة:



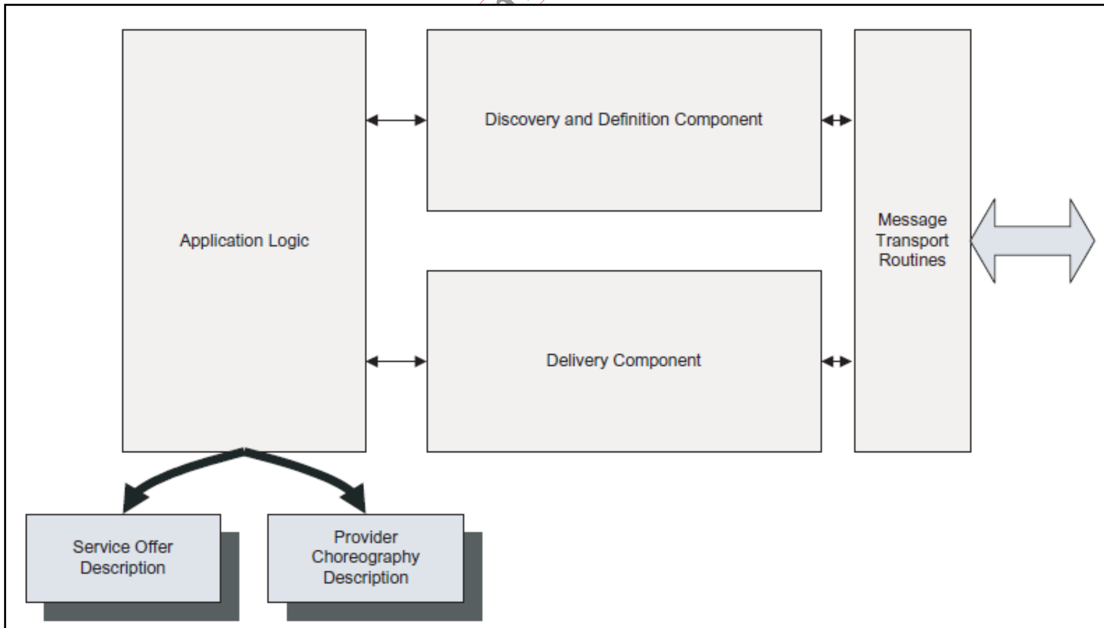
الشكل 4-1، البنية الداخلية لوكيل طالب الخدمة

تتألف هذه البنية من منطق التطبيق أو العمل وهو المسؤول عن اتخاذ القرار الخاص باختيار الخدمة وكيفية استخدامها. قد يتطلب ذلك التواصل مع أنظمة اخرى متوافرة لدى المنظمة التي ينوب عنها وكيل طالب الخدمة. وقد يستخدم هذا المكوّن المعلومات المتوافرة في قاعدة معارف المنظمة. في حالات أخرى يتم تحديد منطق التطبيق من قبل المستخدم عن طريق تزويد مجموعة من واجهات الاستخدام التفاعلية، وعندها يتصرف وكيل طلب الخدمة كوكيل عن المستخدم ويقوم باتخاذ القرار بناءً على اختيار المستخدم.

إن أول الأدوار التي يقوم بها منطق التطبيق هو تعريف توصيف متطلبات الخدمة التي يحتاجها. ثم يقوم بتمرير هذا التوصيف إلى مكوّن استكشاف وتعريف الخدمات. إن دور مكوّن استكشاف وتعريف الخدمات هو إدارة بروتوكولات التعريف والاستكشاف وإرسال الرسائل بالطريقة الموصّفة في بنية الـMacro. يتم تحديد شكل ومحتوى هذه الرسائل من خلال مكوّن "lift/lower" الذي يمرر الرسائل بدوره إلى روتين النقل ليتم إرسال الرسالة وفقاً لبروتوكول النقل. غالباً ما يتم استخدام تقنيات الـWSDL في خدمات الوب للتواصل مع مزوّد الخدمة.

عند الحاجة لاتخاذ قرار، مثلاً عند الحاجة لاختيار مزوّد خدمة أو أكثر، يتم تمرير القرار إلى منطق التطبيق ليتم اتخاذه. يقوم مكوّن "lift/lower" بعملية الوساطة على مستوى البيانات، وعندما يستلم الرسائل الواردة يقوم بترجمة محتواها إلى معلومات دلالية وفقاً للانطولوجي ويقوم بتخزينها في قاعدة المعارف، كما ويقوم بتوليد محتوى الرسائل الصادرة باستخدام الحقائق المخزنة في قاعدة المعرفة. وعندما يتم تعريف الخدمة يقوم منطق التطبيق بتمرير عنوان بروتوكول التسليم إلى مكوّن التسليم.

أما البنية الداخلية للوكيل المزوّد للخدمة فيمكن توضيحه بالشكل "4-2":



الشكل 4-2، البنية الداخلية لوكيل مزوّد الخدمة

إن افتراضنا بأن الوساطة على مستوى البروتوكول تتم من قبل طالب الخدمة يجعل بنية مزوّد الخدمة بسيطة.

كما في طالب الخدمة فإن مزوّد الخدمة يمتلك منطق التطبيق أو العمل والذي يكون مسؤول عن تحديد عروض الخدمات التي سيقدمها لطالب الخدمة بالإضافة إلى تزويد الخدمة ذاتها. كذلك كما في طالب الخدمة فقد يتطلب ذلك من مزوّد الخدمة التواصل مع أنظمة أخرى متوافرة لدى المنظمة التي يمثّلها وكيل مزوّد الخدمة. يقوم منطق التطبيق بتحضير توصيف عرض الخدمة ويقوم بتسجيله لدى مزوّد خدمة استكشاف الخدمات. كما ويقوم بتحضير بروتوكول التسليم المرافق للخدمة ويقوم بنشره على الوب مترافقاً من عنوان "URI".

يحتوي مكوّن تعريف الخدمة على واجهة تسمح لطالبي الخدمات بإرسال توصيف متطلبات الخدمات التي يبحثون عنها وغالباً ما تكون باستخدام تقنيات الـWSDL في خدمات الوب، وعند استلام هذه الطلبات يقوم منطق التطبيق لدى مزوّد الخدمة بإعداد قائمة بالخدمات القادر على تزويدها والتي تلبي احتياجات طالب الخدمة. يتم إرسال هذه القائمة إلى طالب الخدمة عن طريق واجهة مكوّن تعريف الخدمات.

في حال تم استلام رسالة اختيار أحد الخدمات من قبل طالب الخدمة عن طريق مكوّن تعريف الخدمات فيقوم بالإجابة برسالة تأكيد والتي تحتوي على عنوان "URI" توصيف بروتوكول التسليم والذي يتم الحصول عليه من قبل منطق التطبيق. ويتم تنفيذ بروتوكول التسليم من قبل مكوّن تسليم الخدمات باستخدام واجهة قد تعتمد على تقنيات الـWSDL في خدمات الوب الدلالي. ويتم إعلام منطق التطبيق بشكل مستمر بالوضع الحالي للمحادثات الجارية مع طالب الخدمة، كما وتتم استشارته عند الحاجة لاتخاذ قرار ما أثناء تنفيذ البروتوكول.

2-4. مقاربات خدمات الوب الدلالية:

تعد خدمات الوب الدلالية من مواضيع الأبحاث الجديدة التي شهدت العديد من الدراسات والأبحاث وقد ظهرت مجموعة من المقاربات والمعايير بهدف إضافة البعد الدلالي إلى توصيف خدمات الوب بما يحولها إلى خدمات وب دلالية ومن أهمها:WSDL-S، OWL-S، METEOR-S .

1-2-4. WSDL-S:

كنا قد عرّفنا مسبقاً الـWSDL بأنها أحد تقنيات توصيف خدمات الوب. يمكن بالاعتماد على القدرات التوصيفية للـWSDL إضافة مفاهيم دلالية لتوصيف متطلبات خدمة الوب. سنتعرف على الـWSDL-S من خلال مثال خدمة طلب شراء، حيث تتم إضافة بعد دلالي لدخل وخرج خدمة طلب الشراء "ProcessPurchaseOrder"، أما المفاهيم الدلالية والعلاقات فتتم نمذجتها باستخدام أنطولوجية الـOWL، وبذلك يكون ملف توصيف طلب الشراء PurchaseOrder.wsdl بالشكل التالي:

```
<definitions name="PurchaseOrder"
targetNamespace="http://www.ourdemos.com/purchaseorder/wsdl/PurchaseOrder/"
xmlns="http://www.w3.org/2004/08/wsdl”
xmlns:tns="http://www.ourdemos.com/purchaseorder/wsdl/PurchaseOrder/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsd1=”http://www.ourdemos.com/purchaseorder/”
xmlns:wssem="http://www.ibm.com/xmlns/WebServices/WSSemantics"
xmlns:POOntology="http://www.ibm.com/ontologies/PurchaseOrder.owl">
<types>
<xs:import namespace=" http://www.ibm.com/xmlns/WebServices/WSSemantics"
```

```
schemaLocation= "WSSemantics.xsd />
<xs:import namespace=" http://www.ourdemos.com/purchaseorder/"
schemaLocation= "POBilling.xsd />
<xs:import namespace=" http://www.ourdemos.com/purchaseorder/"
schemaLocation= "POItem.xsd />
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace=" http://www.ourdemos.com/purchaseorder/wsdl/PurchaseOrder/"
xmlns="http://www.ourdemos.com/purchaseorder/wsdl/PurchaseOrder/">
<!—Semantic annotations for these complex types are given in their respective type
definitions>
<xs:complexType name="processPurchaseOrderRequest">
<xs:all>
<xs:element name="billingInfo" type="xsd1:POBilling"/>
<xs:element name="orderItem" type="xsd1:POItem"/>
</xs:all>
</xs:complexType>
<!—Semantic annotation is added directly to non-leaf element />
<xs:element name= "processPurchaseOrderResponse" type="xs:string
wssem: modelReference ="POOntology#OrderConfirmation"/>
</xs:schema>
</types>
<interface name="PurchaseOrder">
<!—Category is added as an extensible element of an interface>
<wssem:category name= “Electronics” taxonomyURI=”http://www.naics.com/”
taxonomyCode=”443112” />
<operation name="processPurchaseOrder” pattern=wsdl:in-out>
<input messageLabel = ”processPurchaseOrderRequest"
element="tns:processPurchaseOrderRequest"/>
<output messageLabel ="processPurchaseOrderResponse"
element="processPurchaseOrderResponse"/>
<!—Precondition and effect are added as extensible elements on an operation>
<wssem:precondition name="ExistingAcctPrecond"
wssem:modelReference="POOntology#AccountExists">
<wssem:effect name="ItemReservedEffect"
wssem:modelReference="POOntology#ItemReserved"/>
</operation>
</interface>
</definitions>
```

ويتم تعريف الإضافات على المستند السابق بصيغة xsd بالشكل التالي:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.ibm.com/xmlns/stdwip/Web-services/WS-Semantics"
xmlns:wssem="http://www.ibm.com/xmlns/stdwip/Web-services/WSSemantics"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
<attribute name="modelReference" type="anyURI" use="optional"/>
```

```
<attribute name="schemaMapping" type="anyURI" use="optional"/>
<element name="category" maxOccurs="unbounded">
<complexType>
<complexContent>
<extension base="wsdl:documented">
<attribute name="categoryname" type="NCName" use="required""/>
<attribute name="taxonomyURI" type="anyURI" use="required""/>
<attribute name="taxonomyValue" type="String" use="optional"/>
<attribute name="taxonomyCode" type="integer" use="optional"/>
</extension>
</complexContent>
</complexType>
</element>
<element name = "precondition">
<complexType>
<complexContent
<restriction base="anyType">
<xsd:attribute name="name" type="string" />
<attribute name="modelReference" type="anyURI" />
<attribute name="expression" type="string" />
</restriction>
</complexContent>
</complexType>
</element>
<element name="effect">
<complexType>
<complexContent
<restriction base="anyType">
<xsd:attribute name="name" type="string" />
<attribute name="modelReference" type="anyURI" />
<attribute name="expression" type="string" />
</restriction>
</complexContent>
</complexType>
</element>
</schema>
```

4-2-2. OWL-S:

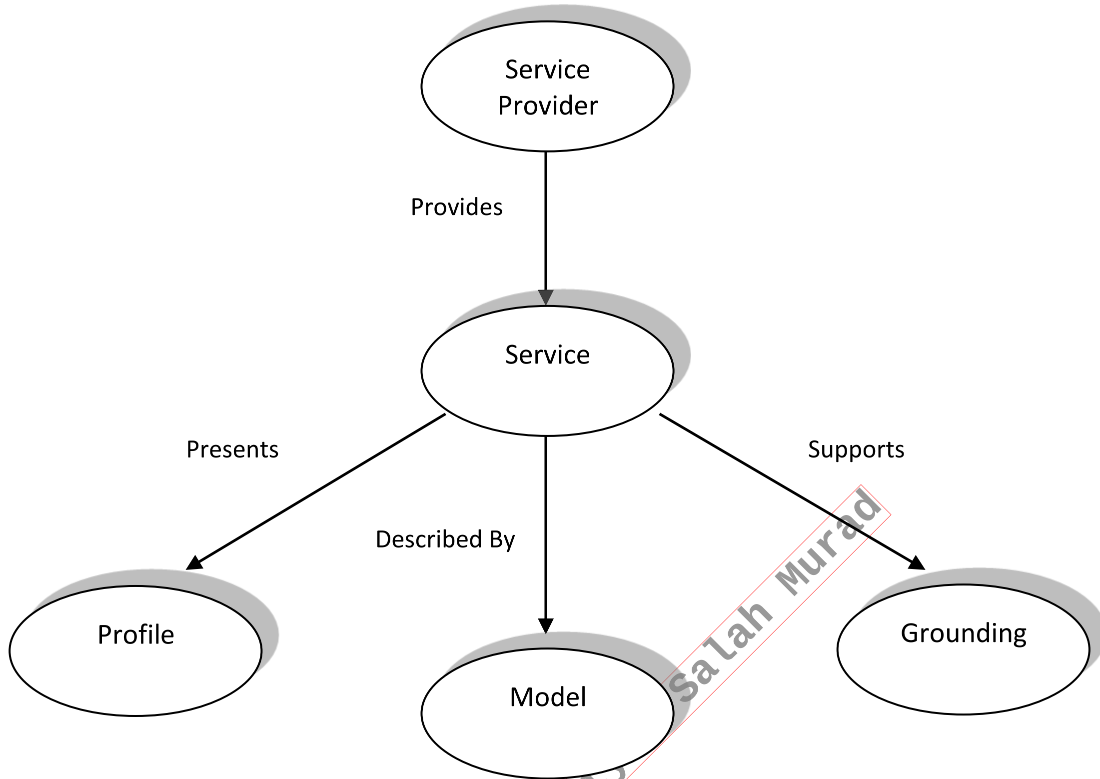
تقوم الـOWL-S بتعريف الأنطولوجية الخاصة بالخدمات من خلال أربعة عناصر أساسية كما في الشكل التالي "4-3" وهي:

أ- الخدمة: ويعبر عن المفهوم الذي يمثّل خدمة الوب وكل خدمة هي مثال عن هذا المفهوم.

ب- ملف الخدمة: يقوم بتوصيف الهدف من الخدمة بشكل عام، موضّحاً المواصفات الوظيفية وغير الوظيفية للخدمة والتي يجري استخدامها لتحديد الخدمة من خلال وصفها الدلالي.

ت- نموذج الخدمة: يقوم النموذج بتوصيف كيف تقوم الخدمة بتأدية وظيفتها بما في ذلك توصيف تفصيلي عن عملياتها الداخلية.

ث- ملف تنفيذ الخدمة: ويقوم بتوصيف كيفية استخدام الخدمة وكيف يمكن لطالبي الخدمة استدعاؤها.



الشكل 4-3، أنطولوجية الـOWL-S

يوضح المخطط السابق الأنطولوجية الخاصة بنموذج الـOWL-S والذي يعتبر أحد أهم نماذج خدمات الوب الدلالي، حيث يربط مفهوم الخدمة بين ملف الخدمة (Service Profile)، نموذج الخدمة(Service Model)، وملف توصيف تنفيذ الخدمة (Service Grounding) من خلال مجموعة من المواصفات وهي على التوالي: تقدم (*Presents*)، توصف من خلال(*DescribedBy*)، وتدعم (*Supports*). سنستعرض مكوّنات هذه الأنطولوجية من خلال خدمة حجز الطيران.

1- الخدمة (Service):

وهي الوظيفة التي يقدمها طرف ما "Provider" لطرف أخر "Requester"، بحيث يتم توصيف الهدف منها، كيفية عملها، وكيفية تنفيذها. تكون هذه الوظيفة قابلة للإستدعاء عن طريق الوب باستخدام عنوان محدد لهذه الخدمة.

2- ملف الخدمة (Service Profile):

يستخدم ملف الخدمة لتوصيف الهدف المنشود من الخدمة، موصّفاً الخدمة المقدمة من قبل مزوّد الخدمة والخدمة المطلوبة من طالب الخدمة.

2-1. هرمية ملف الخدمة (*Profile Hierarchy*):

من الممكن وضع ملف الخدمة ضمن بنية من مجموعة ملفات (Profiles)، وكمثال يمكن تصنيف خدمة الحجز(Reservation Service) كخدمة لحجوزات الطيران (Airline ticketing service)

ضمن هرمية من مجموعة ملفات تتعلق بخدمات التجارة الإلكترونية (e-Commerce service profiles) بالشكل التالي:

```
<owl:Class rdf:ID="AirlineTicketing ">
<rdfs:subClassOf rdf:resource="#E Commerce"/>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#merchandise"/>
<owl:allValuesFrom rdf:resource="#CommercialAirlineTravel"/>
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

في هذا المثال يتم تعريف ملف خدمات حجوزات الطيران كصف جزئي (Subclass) من ملفات التجارة الإلكترونية، حيث المنتج التجاري هو رحلات الطيران بالدرجة الاقتصادية.

2-2. معلومات عن الخدمة (*Service Name*، *Contact*، *Description and Category*):
يحتوي ملف الخدمة على معلومات سهلة القراءة، من خلال مجموعة من المواصفات وهي:

- اسم الخدمة(*serviceName*): وهي عبارة عن خاصية نصية لها قيمة واحدة على الأكثر.
- وصف نصي(*textDescription*): وهي عبارة عن خاصية نصية لها قيمة واحدة على الأكثر.
- معلومات الاتصال(contactInformation): وهي عبارة عن خاصية من نمط الصف *Actor* وتحتوي على مجموعة المعلومات التي لا تتعلق بوظيفة الخدمة(non-functional properties) مثل الاسم، رقم الهاتف، الفاكس، أوعنوان البريد الإلكتروني وغيرها من المعلومات. كما من الممكن أن يتم تحديد تصنيف للخدمة(Service Category)، على الرغم من غياب وجود طريقة تصنيف(Classification Scheme) ثابتة ومحددة. يوضح مثالنا عن خدمة حجوزات الطيران هذه المعلومات بالشكل التالي:

```
<profile:serviceName>BravoAirReservationAgent</profile:serviceName>
<profile:textDescription>This service...</ profile:textDescription >
<profile:contactInformation>
<actor:Actor rdf:ID="BravoAir-reservation">
<actor:name>BravoAir Reservation department</actor:name>
<actor:phone>412 268 8780</actor:phone>
<actor:email>Bravo@Bravoair.com</actor:email>
[...]
</actor:Actor>
</profile:contactInformation>
<profile:serviceCategory>
<addParam:UNSPSC rdf:ID="UNSPSC-category">
<profile:value>Travel Agent</profile:value>
<profile:code>90121500</profile:code>
</addParam:UNSPSC>
```

```
</profile:serviceCategory>
[...]
```

2-3. بارامترات الملف (*Profile Parameter*):

بالإضافة إلى مجموعة المعلومات المذكورة سابقاً، يمكن لملف الخدمة أن يحتوي على قائمة من الخواص التي لا تتعلق بوظيفة الخدمة (non-functional properties) الغير محددة بمجال محدد، وتعرف ببارامترات الخدمة (Service Parameters)، فعلى سبيل المثال يمكن أن نضيف المعلومات التالية لخدمة حجوزات الطيران:

```
<profile:serviceParameter>
<addParam:GeographicRadius rdf:ID="BravoAir−geographicRadius">
<profile:serviceParameterName>BravoAir Geographic
Radius</profile:serviceParameterName>
<profile:sParameter rdf:resource="Country.owl#UnitedStates" />
</addParam:GeographicRadius>
</profile:serviceParameter>
```


2-4. توصيف وظيفة الخدمة (*Functionality Description*):

يحدد ملف الخدمة الوظيفة (functionality) التي يقدمها مزوّد الخدمة، من خلال تعريف مدخلات، مخرجات ونتيجة الخدمة (Inputs،Output، and Result). حيث توصف المعلومات التي تتطلبها الخدمة للتنفيذ بالإضافة إلى المعلومات التي يتم توليدها من قبل الخدمة ذاتها.

إن تعريف المدخلات والمخرجات يتم عملياً في نموذج الخدمة (Service Model) وتتم الإشارة إليه من خلال ملف الخدمة من خلال الخواص {*hasInput*، *hasOutput*، *hasResult*} . في مثال خدمة حجوزات الطيران من الممكن إيجاد التوصيف التالي في ملف الخدمة:

```
<profile:hasInput rdf:resource="BravoAirProcess.owl#DepartureAirport"/>
<profile:hasInput rdf:resource="BravoAirProcess.owl#ArrivalAirport"/>
<profile:hasInput rdf:resource="BravoAirProcess.owl#OutboundDate"/>
<profile:hasInput rdf:resource="BravoAirProcess.owl#InboundDate"/>
<profile:hasInput rdf:resource="BravoAirProcess.owl#RoundTrip"/>
[...]
<profile:hasResult rdf:resource="BravoAirProcess.owl#HaveSeatResult"/>
```

بينما يتم تعريف هذه المحددات (المدخلات، المخرجات، والنتيجة) من خلال نموذج الخدمة(Service Model) الذي سنتكلم عنه لاحقاً وذلك بالشكل التالي:

```
<process:Input rdf:ID="DepartureAirport">
<process:parameterType
rdf:datatype="&xsd;anyURI">Concepts.owl#Airport</process:parameterTy
pe>
</process:Input>
<process:Input rdf:ID=" ArrivalAirport ">
<process:parameterType
```

```
rdf:datatype="&xsd;anyURI">Concepts.owl#Airport</process:parameterTy
pe>
</process:Input>
[...]
<process:Output rdf:ID="FlightsFound">
<process:parameterType
rdf:datatype="&xsd;anyURI">Concepts.owl#FlightList
</process:parameterType>
</process:Output>
<process:Output rdf:ID="ReservationID">
<process:parameterType rdf:datatype="&xsd;anyURI">
Concepts.owl#ReservationNumber
</process:parameterType>
</process:Output>
[...]
```

```
<process:hasResult>
<process:Result>
<process:hasResultVar>
<process:ResultVar rdf:ID="CreditLimH">
<process:parameterType rdf:resource="&ecom;#Dollars"/>
</process:ResultVar>
</process:hasResultVar>
<process:inCondition expressionLanguage="&expr;#KIF"
rdf:dataType="&xsd;#string">
(and (current−value (credit−limit ?CreditCard)
?CreditLimH)
(>= ?CreditLimH ?purchaseAmt))
</process:inCondition>
<process:withOutput>
<process:OutputBinding>
<process:toParam rdf:resource="#ConfirmationNum"/>
<process:valueFunction rdf:parseType="Literal">
<cc:ConfirmationNum xsd:datatype="&xsd;#string"/>
</process:valueFunction>
</process:OutputBinding>
</process:withOutput>
<process:hasEffect expressionLanguage="&expr;#KIF"
rdf:dataType="&xsd;#string">
(and (confirmed (purchase ?purchaseAmt) ?ConfirmationNum)
(own ?objectPurchased)
(decrease (credit−limit ?CreditCard)
?purchaseAmt))
</process:hasEffect>
```

```
</process:Result>
</process:hasResult>
```

3- نموذج الخدمة (Service Model):

ويتم من خلاله تحديد كيفية عمل الخدمة من خلال توصيف الوظيفة التي تقدمها الخدمة (functional properties)، حيث ينظر للخدمة كـProcess لتوصيف كيف يمكن التفاعل مع هذه الخدمة.

1-3 توصيف وظيفة الخدمة (*Functionality Description*):

حيث يتم تعريف الخواص {*hasInput*، *hasOutput*، *hasResult*} كما ذكرنا سابقاً ويتم النظر للخدمة كعملية قابلة للاستدعاء، ويتم تنفيذها بخطوة واحدة من وجهة نظر طالب الخدمة. لذلك يجب تحديد مدخلاتها ومخرجاتها. ففي المثال التالي يمكن إيجاد عمليتين، الأولى للحصول على معلومات تفصيلية عن رحلة أما الثانية فلإختيار رحلة متاحة:

```
<process: Process rdf:ID="GetDesiredFlightDetails">
<process:hasInput rdf:resource="#DepartureAirport"/>
<process:hasInput rdf:resource="#ArrivalAirport"/>
<process:hasInput rdf:resource="#OutboundDate"/>
<process:hasInput rdf:resource="#InboundDate"/>
<process:hasInput rdf:resource="#RoundTrip"/>
<process:hasOutput rdf:resource="#FlightsFound"/>
</process: Process>
```

```
<process: Process rdf:ID="SelectAvailableFlight">
<process:hasInput rdf:resource="#FlightsAvailable" />
<process:hasOutput rdf:resource="#SelectedFlight" />
</process: Process>
```

4- تنفيذ الخدمة (Service Grounding):

يتم من خلال ملف توصيف تنفيذ الخدمة وصف كيفية الوصول للخدمة، والانتقال من التجريد إلى التوصيف التقني للخدمة. يتم ذلك من خلال المواصفة "*Supports*".

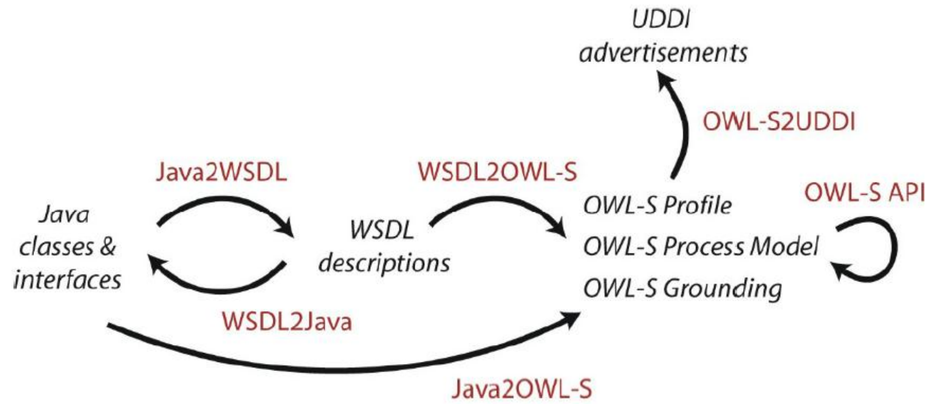
ويمكن للخدمة أن تضم أكثر من ملف توصيف تنفيذ لكن كل ملف يكون مرتبط بعملية جزئية واحدة فقط. ويمكن الوصول إلى ملف التنفيذ الخاص بالمهمة من خلال الوصول إلى نموذج الخدمة أولاً، حيث يتم الربط من خلال هذا التوصيف بين المهام الجزئية والعمليات الموجودة في ملف توصيف الخدمة WSDL ويتم ربط المدخلات والمخرجات الخاصة بالمهام الجزئية بمقابلاتها من العمليات في ملف الـWSDL أيضاً.

3-4. أدوات خدمات الوب الدلالية:

بما أن خدمات الوب الدلالية تقوم بإضافة الوسم الدلالي إلى خدمات الوب فقد ظهرت مجموعة من الأدوات أو الأدوات الصغيرة والتي تقوم بتنفيذ مهام جزئية مثل التوليد الآلي لتوصيف الـWSDL من

خلال صفوف (Classes) مكتوبة بلغة البرمجة الـJava. نعرض فيمايلي بعض هذه الأدوات موضحة بالشكل "4-4":

- Java2WSDL
- WSDL2OWL-S
- Java2OWL-S
- OWL-S2UDDI
- OWL-S API



الشكل 4-4، بعض أدوات الوب الدلالي

4-4. خلاصة:

تم من خلال الفصل إستعراض نموذج خدمات الوب الدلالية والبنية التقنية التي يمكن استخدامها، كما تطرقنا إلى العناصر الأساسية لتوصيف الخدمات، الاستكشاف والوساطة ومكان كل منها في دورة الحياة بالنسبة لاستخدام خدمات الوب الدلالية في أنظمة المعلومات.

الجزء الثالث

تطوير نموذج

نظم المعلومات ذات الخدمات الدلالية

النموذج النظري والرياضي لنظم المعلومات خدمية التوجّه الدلالية

Theoretical and Mathematical Model

مقدّمة:

في بنية النظم خدمية التوجّه والمعتمدة على إعادة استخدام مجوعة من الخدمات المنتشرة لتأمين وظيفة ما، تظهر الحاجة للبحث في مجموعة من المشاكل التي يساعد حلها على إيجاد آفاق تطبيقية وعملية أكثر تعقيداً. إن نظم المعلومات ذات البنى خدمية التوجّه تعتمد في بنيتها على استخدام مجموعة من خدمات الوب المتواجدة عبر الإنترنت حيث يتم نشر وتعميم واجهة التخاطب مع هذه الخدمات بشكل قابل للإستكشاف، بحيث توضّح هذه الواجهة طريقة التفاعل مع الخدمة المقدّمة وتكون بمثابة العقد بين مزوّد الخدمة ومستهلكها.

من الواضح بأن تشكيل هذه البنية يتم غالباً بطريقة يدوية وبمجهود بشري. قد لا يشكل هذا مشكلة بالنسبة لبعض الحالات التي يتم فيها استخدم نفس الخدمات بعد تجميعها وإعدادها من أول مرة، لكن بالنسبة لمستهلكي خدمة أخرين قد يكون من الضروري تحديد الخدمات ومعرفة طريقة التخاطب معها بشكل أكثر ديناميكية وبحاجة أقل للتدخل البشري، أي أن نفس الطلب قد يرسل إلى النظام فيقوم النظام في كل مرة بإستخدام مزوّد خدمة مختلف لتلبية هذا الطلب. كما ويمكن أن يؤثر توافر الخدمة المستهلكة على الوظيفة الكليّة للنظام خدمي التوجّه، لذلك سيكون من الضروري أن يتمكن النظام من البحث بشكل أتوماتيكي على خدمة بديلة ليقوم بتسجيلها كمزوّد خدمة واستخدامها لضمان استمرارية العمل.

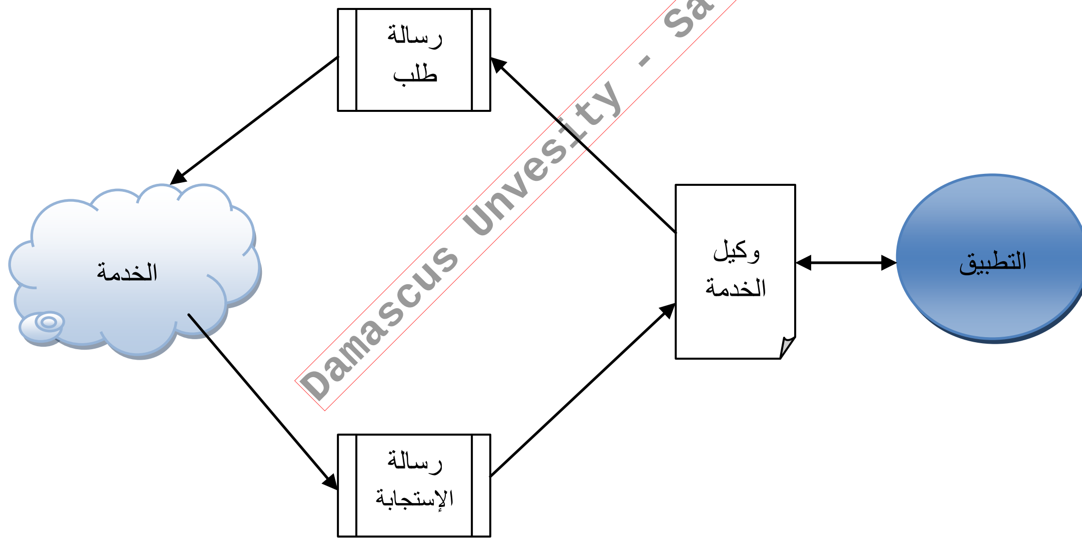
قد يكون من الصعب أحياناً إيجاد الخدمة المطلوبة لبناء النظام خدمي التوجّه بشكل مباشر، في الوقت الذي يكون من الممكن إيجاد تركيبة من تفاعل مجموعة من الخدمات لتوفير الخدمة النهائية المرجوة، وهي ما يمكن أن نسميه بالخدمة الإفتراضية أو المركّبة.

لعل أهم التحديات التي قد تواجهنا في حل مشاكل من هذا النوع هو أن تكون الخدمات المتوافرة مفهومة بالنسبة للآلة، وذلك حتى يتمكن النظام من إيجادها، ويحدد طريقة التخاطب معها، وبالتالي تقديم الخدمة النهائية المرجوة، ومن هنا ندرك أهمية وجود الوب الدلالي كتطوّر مهم في عالم تقنيات الوب كمفهوم يساعد على التغلب على هذا النوع من المشاكل.

1-5. بناء نظم المعلومات خدمية التوجّه بالطريقة التقليدية:

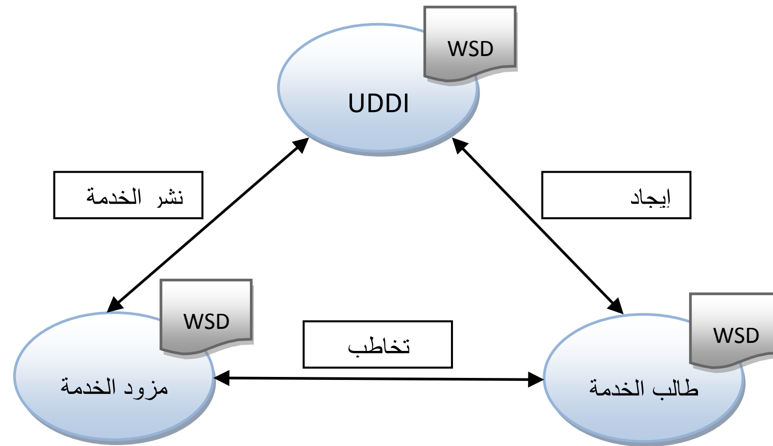
يتم بناء النظم خدمية التوجّه بالإعتماد على خدمة أو مجموعة من الخدمات التي يقدّمها مزوّد خدمة أو أكثر بشكل مجاني أو مدفوع الأجر بدلاً من بناء النظام من الصفر. يتم ذلك بشكل مشابه لما يُعرف بالتطوير المعتمد على المكوّنات بحيث تكون الخدمة بمثابة المكوّن البرمجي الذي يجري إعادة استخدامه. تعتمد التقنيات التي يجري استخدامها في النظم الخدمية التوجّه على وجود وكيل (Proxy) للوظائف التي يقدّمها مزوّد الخدمة عند طالب الخدمة بشكل مسبق كما في الشكل "1-5" ولقد تعدّدت هذه التقنيات وتطوّرت ومنها:

1- إستدعاء الإجرائيات عن بعد (RPC)؛
2- Common Object Request Broker Architecture (CORBA)؛
3- Sun's Enterprise Java Beans؛
4- Microsofts DCOM؛
5- Remote Method Invocation – RMI؛
6- Net Remoting.



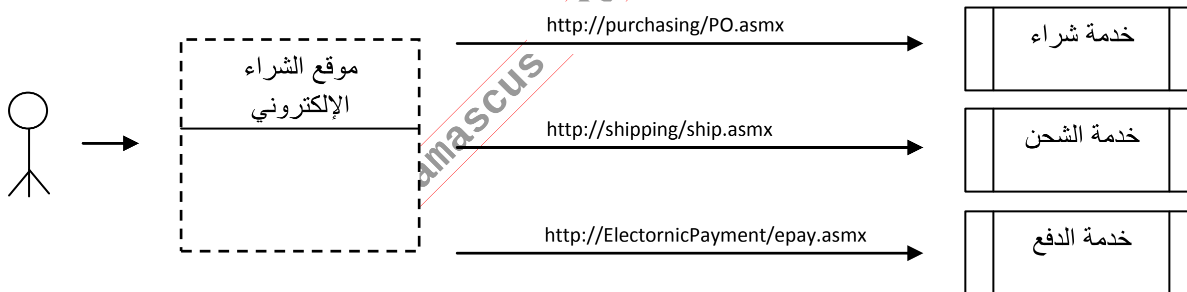
الشكل 5-1، مستهلك الخدمة يعتمد على وجود وكيل للخدمة

على الرغم ممّا توفّره التقنيات السابقة من إمكانيات لبناء نظم موزّعة وخدمية التوجّه إلا أنها امتازت بعدد من المساوئ ومنها: التعقيد في بروتوكولات التواصل، نقص في المقاييس والمعايير الموحّدة، وعدم دعم كل تقنية من قبل الشركات المنافسة. ومن هنا فيمكن النظر إلى ظهور خدمات الوب بمثابة الحل لبناء النظم خدمية التوجّه وذلك لأنها تستخدم بروتوكول الإنترنت لتبادل الرسائل، وبطريقة معيارية باستخدام الـ WSDL المعتمد على الـXML كما هو موضّح في الشكل "2-5". لذلك رأى الباحثون بأن خدمات الوب تساعد على انتشار واعتماد النظم ذات البنى خدمية التوجّه بشكل أوسع.



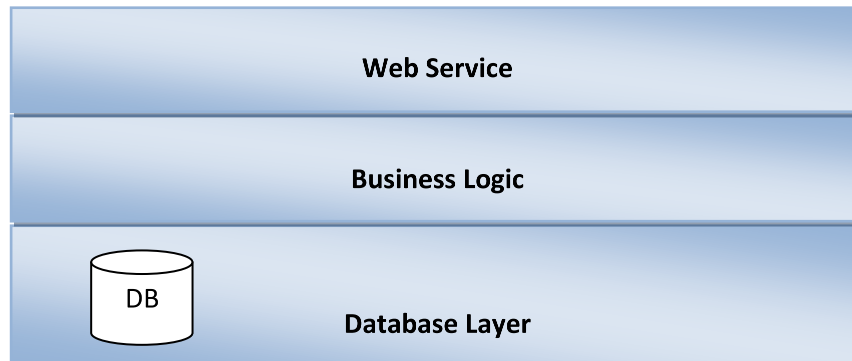
الشكل 5-2، العلاقة بين طالب الخدمة ومزوّد الخدمة في خدمات الوب

لنأخذ مثال المخزن الإفتراضي والذي يتم فيه تقديم خدمة الشراء الإلكتروني لمجموعة من السلع عن طريق موقع متخصص، يمكن للزبون طلب كميات معينة من مادة أو مجموعة مواد، يختار طريقة ومكان وموعد الشحن بالإضافة لطريقة ومعلومات الدفع كما هو موضّح بالشكل "5-3". من أجل بناء هذا النظام يقوم مهندس البرمجيات بالبحث في الانترنت ومن خلال مواقع البحث عن خدمات الوب UDDIs عن مزوّدي ثلاث خدمات وهي خدمة شراء مواد، وخدمة شحن البضائع، وخدمة الدّفع الإلكتروني. بعد العثور على هذه الخدمات يقوم بإضافة وكلاء هذه الخدمات إلى النظام الذي يقوم ببنائه، ثم يتم استدعاء هذه الخدمات من خلال عناوينها المحددة في سياق هذا النظام.



الشكل 5-3، نظام الشراء الإلكتروني

إن كل خدمة من الخدمات السابقة تتمتع بالبنية الموضّحة بالشكل "5-4".



الشكل 5-4، بنية التطبيق المعتمد على خدمات الوب

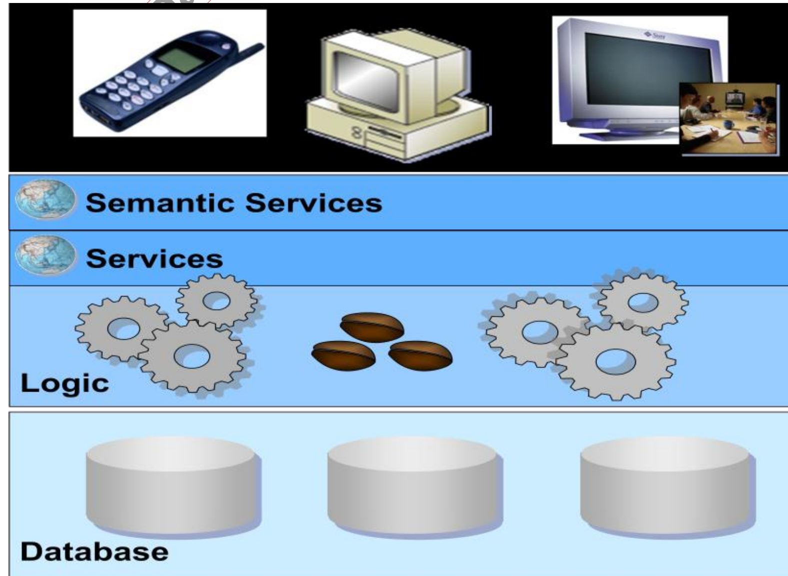
يمكننا استنتاج وتحديد مجموعة من المشاكل التي قد تواجهنا في بناء هذه النظم وهي:

1- العلاقة بين الخدمات محددة بشكل مسبق وثابت وبشكل برمجي؛
2- الربط بين الخدمات يتم بشكلٍ يدويّ ويحتاج لتدخلٍ بشري وبرمجي؛
3- أي تغيير بطريقة تقديم الخدمة لمدخلاتها، لمخرجاتها، لعنوانها، أوبوظيفة الخدمات التي قد تعتمد عليها سيؤدي إلى فشل في وظيفة كلّ الأنظمة التي تعتمد على هذه الخدمة؛
4- صعوبات في تصنيف، توصيف، والبحث عن الخدمات المتوافرة عند الحاجة لبناء أنظمة خدمية التوجّه أوحتى عند الحاجة لبناء خدمات أكثر تعقيداً حيث يتم البحث من خلال مخزن الخدمات عن طريق الـ UDDI؛
5- إن ذذ ببنية خدمية التوجية يقلّل من إمكانية إعادة استخدام الخدمات المركّبة والأكثر تعقيداً والإكتفاء بإمكانية إعادة استخدام الخدمات البسيطة؛
6- إن الطريقة الستاتيكية التي يتم فيها بناء أنظمة ببنية خدمية التوجية يجعلها معتمدة وبشكل مباشر على هذه الخدمات ومزوّديها حصراً؛
7- إن الطريقة الستاتيكية التي يتم فيها بناء أنظمة ببنية خدمية التوجية لا يتيح إمكانية التعامل مع حالات فشل الخدمات المبنية عليها، عدم توافرها، أو أداء الخدمات نفسها.

يمكن التغلب على معظم المشاكل السابقة من خلال مقاربة دلالية يتم من خلالها استخدام الخدمات بالإعتماد على وصفها الدلالي عوضاً عن استخدامها من خلال عنوانها الثابت والمحدد بشكل مسبق.

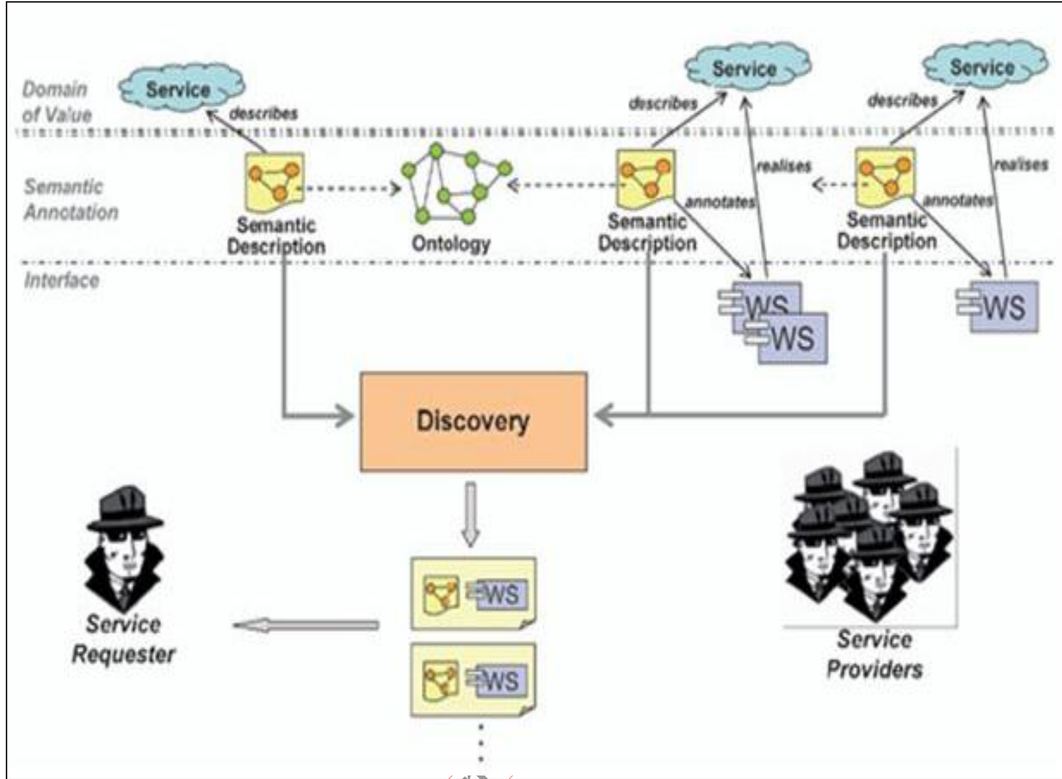
2-5. المقاربة الدلالية لاستخدام الخدمات في نظم المعلومات خدمية التوجّه:

إن إضافة البعد الدلالي لبنى التطبيقات تتم بإضافة طبقة جديدة تمثل الوصف الدلالي للخدمات والتي تقوم بتغليف طبقة الخدمات الأساسية التي يقدّمها التطبيق كما هو موضّح في الشكل "5-5":



الشكل 5-5، بنية التطبيق المعتمد على الخدمات الموصفة دلالياً

إن عملية تقديم الخدمة ستتطلب من مزوّد الخدمة إضافة توصيف لهذه الخدمة، هذا التوصيف يتم بشكل دلالي بالإستعانة بالأنطولوجية، ليتم الوصول إلى الخدمة والبحث عنها من خلال هذا التوصيف والذي يؤثّر في عملية تخزين واستكشاف الخدمات المقدّمة كما في الشكل "6-5":



الشكل 6-5، إستخدام الخدمات الموصفة دلالياً

في الوقت الذي ركّزت فيه معايير خدمات الوب الدلالية ومنها WSDL-S و OWL-S على توصيف واستكشاف خدمات الوب ظهرت عدة مقاربات لاستخدام تقنيتي الوب الدلالي وخدمات الوب وذلك بهدف:

1- البحث عن خدمات الوب وتصنيفها، كحل لمشكلة البحث عن المكوّنات اللازمة لبناء البرمجيات.
2- إستخدام الخدمات الدلالية في بناء النظم التي لا تمتلك بنى عمليات "Process Model"، أي أنها تقدم مجموعة من الخدمات التي لا يرتبط بعضها مع بعض.
3- بناء النظم بالإعتماد على الخدمات من خلال البحث عن الخدمات المطابقة للوصف الدلالي ولكن بالإعتماد على تصميم محدّد برمجياً لطريقة تفاعل الخدمات. تتميز هذه الطريقة بالحاجة للتدخل البشري والبرمجي بشكل مستمر والتكلفة المرتفعة لصيانة النظم وإدارتها وعدم إمكانية إعادة إستخدام الوظيفة الجديدة التي يُقدّمها النظام.
4- بناء النظم من خلال التوجيه المستمر من قبل المستخدم لمسار التنفيذ. تتميز هذه الطريقة بالإعتماد بشكل كلّي على المستخدم العادي والذي يفتقر عادةً للخبرة والإمكانيات المطلوبة، كما لا يوجد ضمان بالوصول إلى الهدف النهائي للمستخدم، إلى جانب عدم إمكانية إعادة إستخدام الوظيفة الجديدة التي يقدّمها النظام.

سنقوم بدورنا بتوظيف تقنيتي الوب الدلالي وخدمات الوب من خلال توصيف إطار عمل لاستخدام الخدمات الموصّفة دلالياً في عملية بناء نظم المعلومات، سنطلق عليه اختصاراً **S_3OS** وهو نموذج نظم

المعلومات خدمية التوجّه الدلالية **S**emantic **S**ervices in **S**ervice **O**riented information **S**ystems . سنعتمد في إطار العمل على الأنطولوجية OWL-S حيث سنقوم بتوسعتها لإضافة مجموعة المفاهيم والعلاقات الضرورية لتمثيل الوظيفة المطلوبة كخدمة مركّبة ترتبط فيها مجموعة من الخدمات دلالياً وتُمثِّل أحد سيناريوهات المكاملة الإفتراضية لأحد مكوناتها، وذلك بهدف تقليل حجم التدخل البشري والبرمجي لبناء وإدارة وصيانة هذه النظم إلى جانب إمكانية الوصول إلى الوظيفة الجديدة باعتبارها خدمة يمكن استدعاؤها ممّا يعزّز إعادة استخدام هذه الوظائف أيضاً.

3-5. النموذج النظري لنظم المعلومات خدمية التوجّه الدلالية (S_3OS):

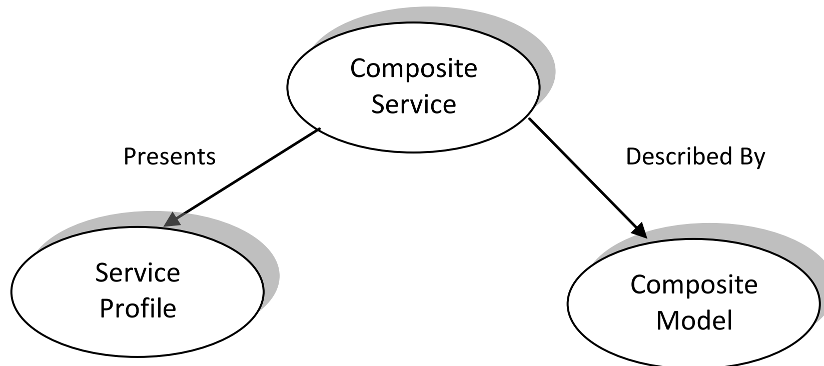
بالإعتماد على الأنطولوجية الخاصة بأحد أهم نماذج خدمات الوب الدلالية، النموذج OWL-S، الموضّح بالشكل "3-4"، والذي قمنا بدراسته سابقاً في الفصل الرابع. حيث يتم الربط بين مفهوم الخدمة (Service)، وملف الخدمة (Service Profile)، ونموذج الخدمة (Service Model)، وملف توصيف تنفيذ الخدمة (Service Grounding) من خلال مجموعة من المواصفات وهي على التوالي: تقدم (*Presents*)، توصّف من خلال (*DescribedBy*)، وتدعم (*Supports*).

سنقوم فيما يلي باستخدام هذه الأنطولوجية وتوسعتها من خلال إضافة مجموعة من المفاهيم التي نسعى من خلالها إلى الوصول إلى توصيف كيفيّة بناء الخدمات ضمن بيئة أنظمة المعلومات المعتمدة على الخدمات ضمن إطار عمل عام.

تعاريف:

- **الخدمة المركّبة (Composite Service):**

وهي عبارة عن خدمة يتم توصيفها من خلال نموذج خدمات مركب (*Composite Model*)، وتمثل من خلال ملف خدمة (*Profile*) كما هو موضّح بالشكل "5-7". تعتمد هذه الخدمات على تركيب مجموعة من الخدمات الموجودة أو الإفتراضية بهدف خلق خدمات جديدة تقدّم وظائف جديدة، أو بهدف زيادة موثوقيّة الحصول على بعض الخدمات، أوتحسين أداء البعض الاخر، ممّا يساعد في بناء خدمات أكثر تعقيداً من حيث الوظيفة باتجاه بناء منظومات خدمية التوجّه بالإعتماد على خدمات وب مُوصّفة دلالياً.



الشكل 5-7، نموذج الخدمة المركّبة

- **نموذج الخدمة المركّب (Composite Service Model):**

ويتم من خلالها تحديد كيفيّة عمل الخدمة المركّبة من خلال توصيف الوظيفة التي تقدمها الخدمة (functional properties)، حيث يُنظر للخدمة المركّبة كمجموعة من العمليات (Processes) التي لا يتم تعريفها بأسمائها الصريحة بشكل مباشر وإنما من خلال وصف وظيفتها والهدف منها ومجال عملها، وهكذا تمثل كل عملية طالب خدمة افتراضي (Process as a requester) وذلك من خلال استخدام كل عملية جزئية لما يسمّى بنموذج تفويض الطلب (Request Delegation Form) وهو ما سيتيح لنا مكاملة الخدمات بطريقة ديناميكية وأكثر فاعلية، كما ويتم ربط هذه العمليات فيما بينها وتبادل البيانات من خلال بارامترات دخلها وخرجها ويتم التمييز بين البارامترات الإختيارية والإجبارية لأداء الخدمة لوظيفتها، ويتم ربط هذه الخدمات بترتيب تنفيذ معيّن بهدف الوصول إلى النتيجة النهائية المرجوّة من الوظيفة التي تقدّمها الخدمة الجديدة.

لتكون هذه العملية شفافة بالنسبة لطالب الخدمة المركّبة يتم بناء إطار العمل بحيث يحتوي على مكامل الخدمات الذي سيتم التطرّق إلى دوره والوظائف التي يتيحها ضمن إطار العمل المقترح لاحقاً. وهكذا يمكن التفاعل مع الخدمة الجديدة الناتجة عن طريق هذا المكوّن والذي يمكن اعتباره أساساً كخدمة أو مجموعة من الخدمات وظيفتها والهدف منها مكاملة الخدمات وما يتطلبه ذلك من عمليات أخرى.

- توصيف وظيفة الخدمة (*Functionality Description*):

حيث يتم تعريف الخواص *{hasInput*، *hasOutput*، *hasResult}* على مستوى كل عملية ويتم النظر للخدمة المركّبة كعملية واحدة قابلة للاستدعاء، ويتم تنفيذها بخطوة واحدة من وجهة نظر طالب الخدمة. لذلك يجب تحديد مدخلاتها ومخرجاتها أيضاً بنفس الطريقة. سنوضّح في المثال التالي كيف يتم تقديم وتوصيف خدمة جديدة هدفها حجز رحلة طيران والحجز الفندقي بشكل مباشر كجزء من عملية التخطيط لرحلة (Trip Planning).


```
<Process: CompositeService rdf:ID=”TripPlanningService”>
 < Process:hasInput rdf:resource=”#DepartureAirport”/>
 < Process:hasInput rdf:resource=”#ArrivalAirport”/>
 < Process:hasInput rdf:resource=”#ArrivalDate”/>
 < Process:hasInput rdf:resource=”#DepartureDate”/>
 < Process:hasInput rdf:resource=”#Destination”/>
 < Process:hasInput rdf:resource=”#ReturnDate”/>
 <process:hasInput rdf:resource=”#Class” rdf:resource="Optional"/>
 < Process:hasOutput rdf:resource=”#TripPlanDetails”/>
<process:Process rdf:resource="FlightReservation">
 <process:order>1</process:order>
 <process:RequestDelegation>
    nonFunctionalProperties
    dc#description hasValue"the output of the service is a flight booking."
    endNonFunctionalProperties
    matchType #ExactMatch
    definedBy
    ?BookingIdentifier memberOf #bookingIdentifier and
    ?FlightBookingTicket[bookingTicketIdentificator
    hasValue?BookingTicketIdentificator، airTripInfohasValue? FlightInfo]
    memberOf #flightBookingTicket and
```

```
      ?FlightInfo[ start hasValue?Start،
      departure hasValue?Departure،
      arrival hasValue?Arrival،
      class hasValue?Class]
      memberOf # FlightInfo
  </process:RequestDelegation>
  <process:hasInput rdf:resource=”#DepartureAirport”/>
  <process:hasInput rdf:resource=”#ArrivalAirport”/>
  <process:hasInput rdf:resource=”#TicketReservationDate”/>
  <process:hasOutput rdf:resource=”#ReservationID”/>
</process:Process>
<process:Process rdf:resource="HotelReservation">
  <process:RequestDelegation>
      nonFunctionalProperties
      dc#description hasValue"the output of the service is a hotel booking."
      endNonFunctionalProperties
      matchType #ExactMatch
      definedBy
      ?BookingIdentifier memberOf #bookingIdentifier and
      ?HotelBooking[
      bookingHotelIdentificator hasValue?BookingHotelIdentificator،
      HotelInfohasValue? HotelReservationInfo
      ] memberOf #HotelBooking and
      ? HotelReservationInfo [class hasValue?Class،
      From hasValue?Date،
      To hasValue?Date،
      city hasValue?Location]
      memberOf # HotelReservationInfo
  </process:RequestDelegation>
  <process:order>2</process:order>
  <process:hasInput rdf:resource=”#City”/>
  <process:hasInput rdf:resource=”#StartDate”/>
  <process:hasInput rdf:resource=”#EndDate”/>
  <process:hasInput rdf:resource=”#Class” rdf:resource="Optional"/>
  <process:hasOutput rdf:resource=”#HotelReservationDetails”/>
</process:Process>
< Process:ParameterIntegration>
  <PI>
   <Source rdf:resource="#DepartureAirport "/>
   <Destination rdf:resource="#FlightReservation#DepartureAirport "/>
  </PI>
  <PI>
   <Source rdf:resource="#ArrivalAirport "/>
   <Destination rdf:resource="#FlightReservation#ArrivalAirport "/>
  </PI>
  <PI>
   <Source rdf:resource="#DepartureDate"/>
   <Destination rdf:resource="#FlightReservation#TicketReservationDate"/>
  </PI>
  <PI>
```

```
 <Source rdf:resource="#Destination"/>
 <Destination rdf:resource="#HotelReservation#City"/>
</PI>
<PI>
 <Source rdf:resource="#ArrivalDate"/>
 <Destination rdf:resource="#HotelReservation#StartDate"/>
</PI>
<PI>
 <Source rdf:resource="#ReturnDate"/>
 <Destination rdf:resource="#HotelReservation#EndDate"/>
</PI>
<PI>
 <Source rdf:resource="#Class"/>
 <Destination rdf:resource="#HotelReservation#Class"/>
</PI>
[…]
</ Process:ParameterIntegration>
</ Process:CompositeService>
```

يُمكن النظر إلى الخدمة المركّبة أو نموذج الخدمة المركّبة على أنه جزء من بيان أو قائمة الخدمة (Service Catalogue) التابع لأحد الخدمات البسيطة التي تتألف منها، حيث يمكن بناء سيناريوهات مكاملة متعدّدة لأية خدمة جديدة مُضَافة إلى فضاء العمل أن تُشكّل مجموعة هذه السيناريوهات ما يُمكن تسميته بقائمة الخدمة، وهكذا يُمكن اعتبار المثال السابق كجزء من قائمة خدمة حجوز الطيران (Flight reservation service catalogue) أوجزء من قائمة خدمة حجوز الفنادق (Hotel reservation service catalogue).

- **ملف الخدمة(Service Profile):**

يُستَخدم ملف الخدمة لتوصيف الهدف المنشود من الخدمة المركّبة سواء كانت إفتراضية أو خدمة موجودة على أرض الواقع، مُوصّفاً للخدمة المُقدّمة من قبل مزوّد الخدمة أو مكامل الخدمات والخدمة المطلوبة من طالب أو مُستهلك الخدمة. ويمكن أن يحتوي على:

1- معلومات عن الخدمة (*Service Name، Contact، Description and Category*):
يحتوي ملف الخدمة على معلومات سهلة القراءة، من خلال مجموعة من المواصفات وهي:

أ- إسم الخدمة (*serviceName*): وهي عبارة عن خاصية نصية لها قيمة واحدة على الأكثر.
ب- وصف نصّي (*textDescription*): وهي عبارة عن خاصية نصّية لها قيمة واحدة على الأكثر.
ت- معلومات الاتصال (contactInformation): وهي عبارة عن خاصية من نمط الصف *Actor* وتحتوي على مجموعة المعلومات التي لا تتعلق بوظيفة الخدمة (non-functional properties) مثل الاسم، رقم الهاتف، الفاكس، أوعنوان البريد الإلكتروني وغيرها من المعلومات. يوضّح مثالنا عن خدمة حجوزات الطيران هذه المعلومات بالشكل التالي:

```
<profile:serviceName>BravoAirReservationAgent</profile:service
Name>
<profile:textDescription>This service...</ profile:textDescription >
<profile:contactInformation>
<actor:Actor rdf:ID=”BravoAir−reservation”>
<actor:name>BravoAir Reservation department</actor:name>
<actor:phone>412 268 8780</actor:phone>
<actor:email>Bravo@Bravoair.com</actor:email>
[...]
</actor:Actor>
</profile:contactInformation>
<profile:serviceCategory>
<addParam:UNSPSC rdf:ID=”UNSPSC−category”>
<profile:value>Travel Agent</profile:value>
<profile:code>90121500</profile:code>
</addParam:UNSPSC>
</profile:serviceCategory>
[...]
```

2- بارامترات الملف (*Profile Parameter*):

بالإضافة إلى مجموعة المعلومات المذكورة سابقاً يمكن لملف الخدمة أن يحتوي على قائمة من الخواص التي لا تتعلق بوظيفة الخدمة (non-functional properties) الغير مُحدّدة بمجالٍ محدد، وتُعرف ببارامترات الخدمة (Service Parameters)، فعلى سبيل المثال يمكن أن نُضيف المعلومات التالية لخدمة حجوزات الطيران:

```
<profile:serviceParameter>
<addParam:GeographicRadius
rdf:ID=”BravoAir−geographicRadius”>
<profile:serviceParameterName>BravoAir Geographic
Radius</profile:serviceParameterName>
<profile:sParameter rdf:resource=”Country.owl#UnitedStates” />
</addParam:GeographicRadius>
</profile:serviceParameter>
```

أ. توصيف وظيفة الخدمة (*Functionality Description*):

يُحدد ملف الخدمة الوظيفة (functionality) التي يُقدّمها مزوّد الخدمة، من خلال تعريف مدخلات، مخرجات ونتيجة الخدمة (Inputs،Output، and Result). حيث تُوصَّف المعلومات التي تتطلبها الخدمة للتنفيذ بالإضافة إلى المعلومات التي يتم توليدها من قبل الخدمة ذاتها.

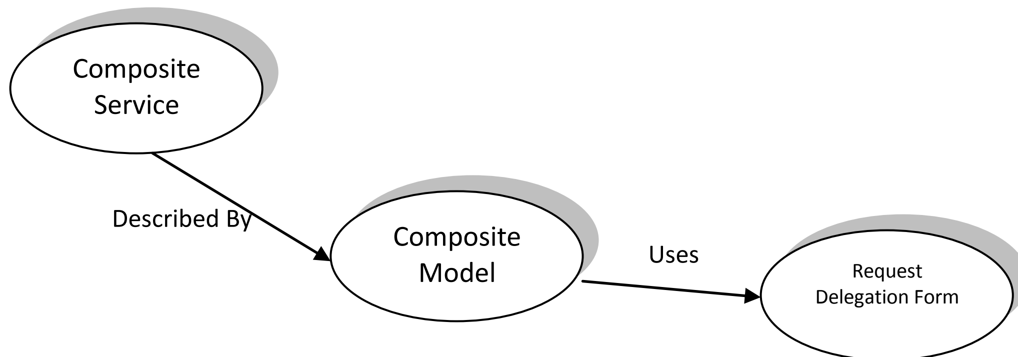
إن تعريف المدخلات والمخرجات يتم عملياً في نموذج الخدمة (Service Model) وتتم الإشارة إليه من خلال ملف الخدمة من خلال الخواص *{hasInput*، *hasOutput*، *hasResult}* . في مثال خدمة حجوزات الطيران من الممكن إيجاد التوصيف التالي في ملف الخدمة:

```
<profile:hasInput
rdf:resource="BravoAirProcess.owl#DepartureAirport"/>
<profile:hasInput
rdf:resource="BravoAirProcess.owl#ArrivalAirport"/>
<profile:hasInput
rdf:resource="BravoAirProcess.owl#OutboundDate"/>
<profile:hasInput
rdf:resource="BravoAirProcess.owl#InboundDate"/>
<profile:hasInput rdf:resource="BravoAirProcess.owl#RoundTrip"/>
[...]
<profile:hasResult
rdf:resource="BravoAirProcess.owl#HaveSeatResult"/>
```

- **نموذج تفويض الطلب (Request Delegation Form):**

يُمثل نموذج التفويض نظرة طالب الخدمة للإمكانيات الوظيفية المطلوبة، والتي يجب أن تحتوي على وصف إمكانيات خدمة الوب المطلوبة. يتم بذلك استخدام نموذج التفويض في توصيف نموذج الخدمة المركّبة لتحويل الخدمة ذاتها إلى طالب خدمة وذلك بهدف إعطاء بعد ديناميكي لمكاملة الخدمات بدلاً من الإشارة الستاتيكية للخدمات البسيطة التي يمكن أن تتألف منها، وهنا تكون مهمة مكامل الخدمات البحث عن الخدمة التي تُحقق المواصفات الوظيفية المذكورة في نموذج التفويض من خلال عملية استكشاف (Discovery) ومطابقة (Matching) للخدمات الموجودة في فضاء العمل بما يتوافق مع درجة المطابقة المُوصّفة في النموذج، وعند إيجاد جميع خدمات الوب المطلوبة المُشكِّلة للخدمة المركّبة يتم استدعاؤها وفقاً للتسلسل والتوصيف الموجودين في نموذج الخدمة المركّبة وكأنها خدمة واحدة موصّفة بشكل ستاتيكي معروف مسبقاً.

يُمكن إيضاح العلاقة بين مفهوم نموذج التفويض ونموذج الخدمة المركّبة والخدمة المركّبة من خلال الشكل التالي "8-5":

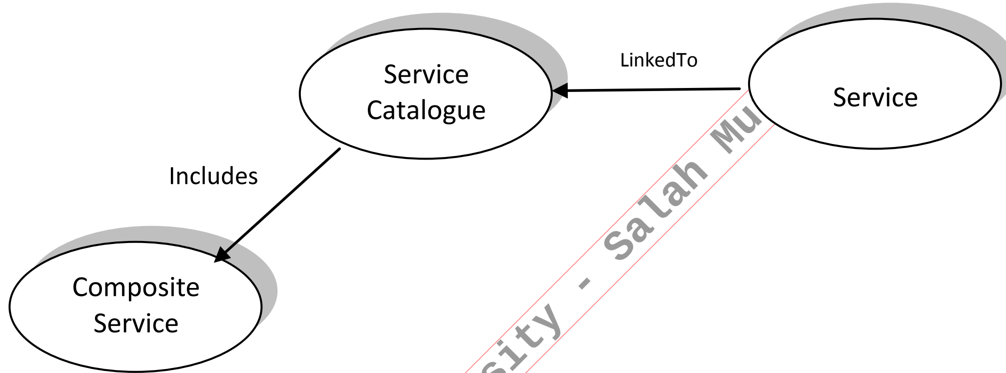


الشكل 5-8، نموذج تفويض الطلب

- **قائمة الخدمة(Service Catalogue):**

وتُمثل مجموعة من السيناريوهات التي يتم فيها استخدام الخدمة كجزء من خدمة ذات وظيفة أكثر تعقيداً، تتكامل فيها مع وظائف مقدّمة من خدمات أخرى يتم التأشير إليها بشكل ستاتيكي من خلال إسم وعنوان الخدمة أو بشكل ديناميكي عن طريق وصف وظيفة الخدمة باستخدام نموذج تفويض الطلب، وذلك بهدف تسهيل بناء منظومات خدمية التوجّه ذات بعد دلالي وديناميكي كما هو موضّح بالشكل "5-9".

كلما ازداد حجم قائمة الخدمة كلما كان من الممكن الإستفادة من الخدمة بشكل أوسع من خلال ازدياد احتمال استدعائها واستخدامها ليس بشكل مباشر وحسب وإنما كجزء من خدمات أكبر أوجزء من منظومات تعتمد على الخدمات في بنائها، ممّا يسهم في تعزيز إعادة إستخدام الخدمة وبناء أنظمة خدمية التوجّه.

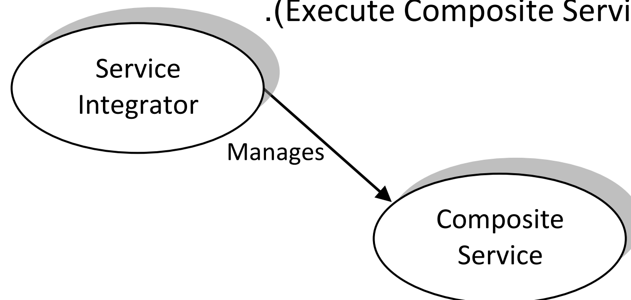


الشكل 5-9، مفهوم قائمة الخدمة

- **مكامل الخدمات:**

إن الوظيفة الأساسية لمكامل الخدمات هي إدارة الخدمات والخدمات المركّبة كما هو موضّح بالشكل "5-10"، حيث يتكّون مكامل الخدمات من مجموعة من خدمات الويب التي ستُتيح لمزوّدي الخدمات إمكانية إضافة خدمات مركّبة بسيناريوهات متعددة للخدمات التي يقومون بتزويدها بما يُشكّل الخدمات المركّبة، كما ويتيح إمكانية الحصول على الخدمات المركّبة واستدعائها وتنفيذها من قبل طالبي الخدمة من خلال مجموعة من الوظائف ومنها:

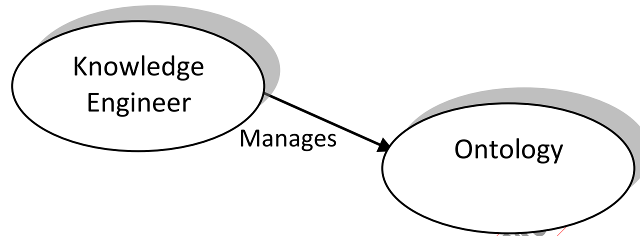
1- إضافة خدمة (Add Service)؛
2- إضافة خدمة مركّبة (Add Composite Service)؛
3- إستخراج خدمة مركّبة (Get Composite Service)؛
4- تنفيذ خدمة مركّبة (Execute Composite Service).



الشكل 5-10، مفهوم مكامل الخدمات

- **مهندس المعرفة:**

إن من أهم الأدوار التي قد يلعبها مهندس المعرفة في النموذج المقدّم هو إدارة الأنطولوجية كما هو موضّح بالشكل "5-11"، حيث يقوم بتحديث قاعدة بيانات الأنطولوجية عند إضافة مفاهيم جديدة، الأمر الذي يتطلب خبير في تمثيل وإدارة المعرفة حتى تتم صياغة المفهوم الجديد بالطريقة الأمثل. حيث يقوم مهندس المعرفة بربط المفاهيم التي تملك نفس الدلالة والمزوّدة من قبل مزوّدي خدمة متعددين بحيث يستطيع مزوّد خدمة ما أن يتعامل مع المفاهيم التي يستخدمها مزوّد خدمة آخر تحمل نفس المعنى.



الشكل 5-11، مفهوم مهندس المعرفة

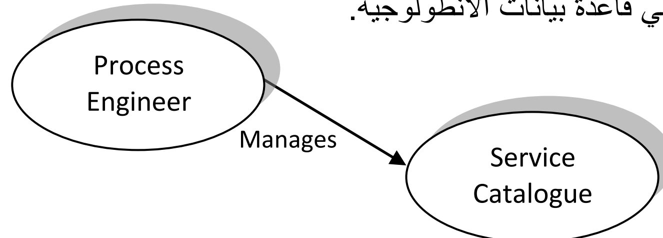
- **الأنطولوجية (Ontology):**

إن كل العمليات السابقة من توصيف للخدمات ومكاملة الخدمات واستكشاف ومطابقة الخدمات وتحديد درجة المطابقة يتطلب وجود أنطولوجية للتسوية بين المفاهيم المختلفة. حيث أن الحديث عن مكاملة مجموعة من الخدمات التي يتم تقديمها من قبل أكثر من مزوّد خدمة يعني بالضرورة إمكانية وجود اختلاف في المفاهيم المُستخدمة في هذه الخدمات وتوصيفها، وهنا يأتي دور الأنطولوجية لفض النزاعات بين المفاهيم، وكلما زاد حجم العمل نكون بحاجة لأنطولوجية أشمل وأكبر حجماً بحيث تكون أكثر قدرة وفاعلية على أداء دور فعال للتعامل مع مفاهيم متعددة من قبل مزوّدي خدمات متعددين، وهذا يضفي بعد دلالي لعملية المكاملة بين الخدمات.

- **مهندس العمليات:**

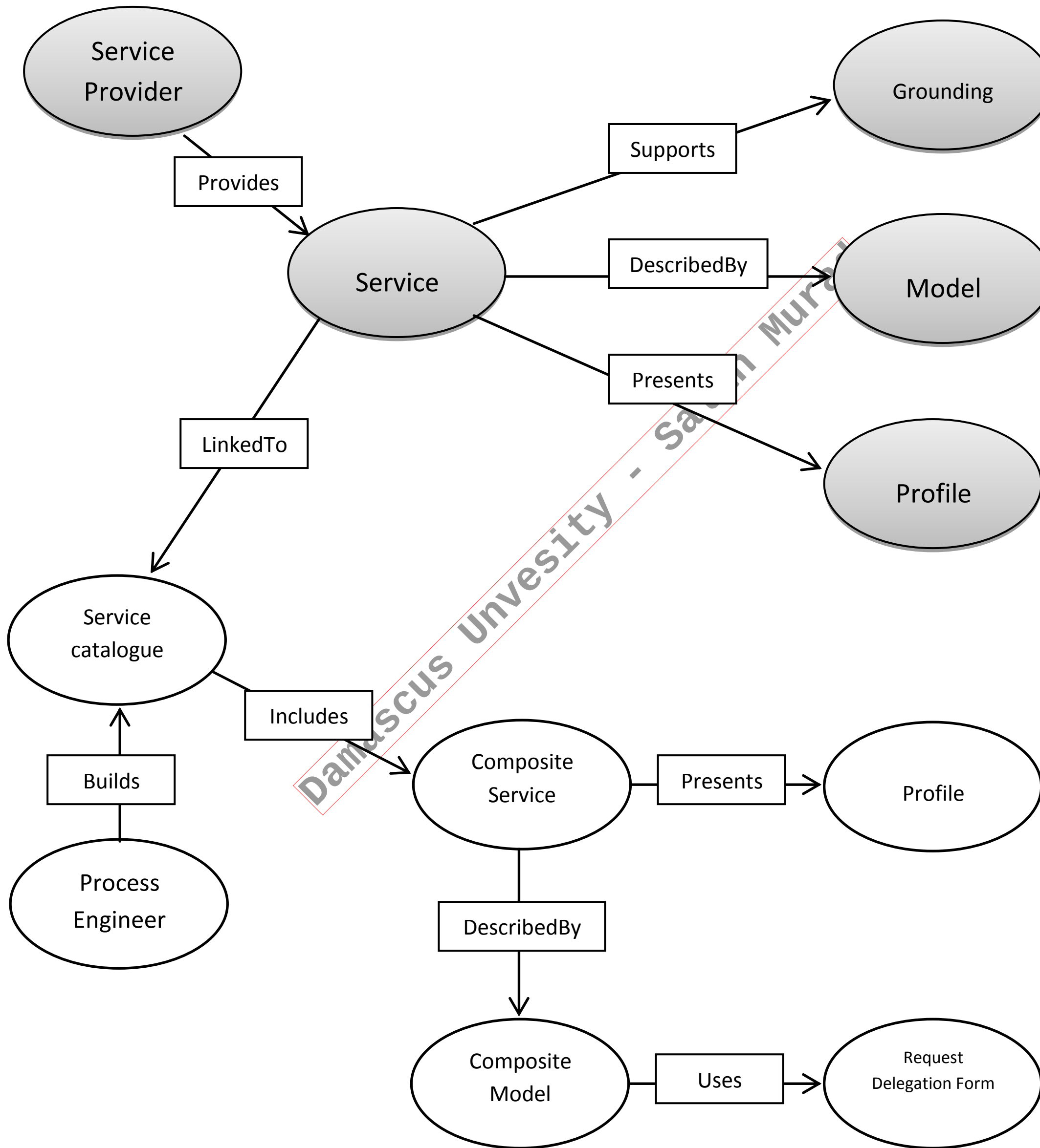
من الممكن إعطاء دور غاية في الأهمية لمهندس العمليات في عملية بناء النظم خدمية التوجّه من خلال إدارة قائمة الخدمة كما هو موضّح بالشكل "5-12"، بحيث لا يقتصر هذا الدور على مزوّدي الخدمات فقط. حيث يقوم مهندس العمليات وهو الخبير في إدراة مهام العمل(Business Processes) وتحسينها (Process Optimization) بإضافة سيناريوهات مكاملة متعددة للخدمات الجديدة المضافة من قبل مزوّدي الخدمات إلى فضاء العمل.

يمكن النظر إلى دور مهندس العمليات في إدارة قائمة الخدمة بشكل مشابه لدور مهندس المعرفة بإدارة المعرفة في قاعدة بيانات الأنطولوجية.



الشكل 5-12، مفهوم مهندس العمليات

من خلال ما سبق يمكننا تقديم النموذج العام من خلال دمج المفاهيم التي تقدمها الأنطولوجية OWL-S بالمفاهيم السابقة التي قمنا بإضافتها والتي تتعلق بتركيب ومكاملة الخدمات، حيث يرتبط مفهوم الخدمة مع قائمة الخدمة من خلال العلاقة *has* والتي ترتبط بدورها مع الخدمات المركّبة من خلال العلاقة *includes* والتي ترتبط بملف الخدمة المركّبة بنفس العلاقة التي تربط الخدمة بملف الخدمة، أما نموذج الخدمة المركّبة فيرتبط من جهة بالخدمة المركّبة من خلال العلاقة *DescribedBy* كما ويرتبط بنموذج تفويض الطلب من خلال العلاقة *Uses* كما هو موضّح من خلال المخطط "13-5":



الشكل 13-5، نموذج نظم المعلومات خدمية التوجّه الدلالي S_3OS

- **مستودع الخدمات:**

يتيح مستودع الخدمات إمكانية نشر خدمات الوب والمعلومات المتاحة عنها إلى جانب توصيفها الدلالي، مع إمكانية تخزين معلومات الخدمات المركّبة الناتجة عن تحديد مجموعة العلاقات بين مجموعة من الخدمات الجزئية التي تتألف منها. بحيث يمكن النظر إلى دور مستودع الخدمات بشكل

مشابه إلى دور الـUDDI في خدمات الوب والتي يتم البحث فيها عن خدمات الوب المتوافرة على الانترنت.

- **مستكشف الخدمات:**

إن الهدف من مستكشف الخدمات هو البحث عن الخدمات الموجودة في مخزن الخدمات بالإعتماد على توصيفها الوظيفي ومطابقتها وتحديد درجة المطابقة.

يوجد العديد من مقاربات إستكشاف الخدمات ومنها المقاربات التالية:

1- المطابقة النصية بالإعتماد على الوصف النصي المُرفق بكل خدمة.
2- المطابقة بالإعتماد على الوصف المنطقي (Logical Description).
3- المطابقة على مستوى بارامترات الدخل والخرج الخاصة بكل من الخدمتين موضوع المطابقة.

سنوضّح هذا من خلال المثال التالي والذي تظهر فيه الخدمة بالتوصيف التالي:

```
nonFunctionalProperties
dc#description hasValue"the output of the service is a flight booking
at a certain class"
endNonFunctionalProperties
definedBy
?BookingIdentifier memberOf #bookingIdentifier and
        ?FlightBookingTicket[
bookingTicketIdentificator hasValue?BookingTicketIdentificator،
airTripInfohasValue? FlightInfo
] memberOf #flightBookingTicket and
?FlightInfo[
  start hasValue?Start،
  departure hasValue?Departure،
  arrival hasValue?Arrival،
  class hasValue?Class
] memberOf # FlightInfo
```

- **المطابقة النصية:**

يمكن لطالب الخدمة الوصول إلى الخدمة الموصَّفة أعلاه من خلال البحث باستخدام النص التالي على سبيل المثال:

"The need is for a service for airline ticket booking from Damascus to Dubai of economy class".

- **المطابقة المنطقية:**

يمكن لطالب الخدمة إستخدام نموذج التفويض التالي للوصول إلى هذه الخدمة:

```
?FlightInfo[
  start hasValue?05June2011،
  departure hasValue?DamascusAirport،
  arrival hasValue?DubaiAirport،
  class hasValue?'economy'
] memberOf # FlightInfo
```

أما إذا كان توصيف الخدمة بالشكل التالي فلن تتم مطابقة نموذج التفويض السابق مع هذه الخدمة:

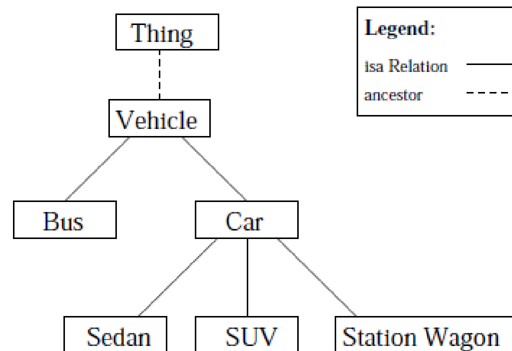
```
?FlightInfo[
  start hasValue?Start،
  departure hasValue?Departure،
  arrival hasValue?Arrival،
  class hasValue?'Business'
] memberOf # FlightInfo
```

السبب في عدم إمكانية مطابقة الخدمة مع نموذج التفويض السابق لكون الخدمة موجّهة لحجوزات تذاكر الطيران من درجة رجال الأعمال فقط، أما طالب الخدمة فهو بحاجة للحجز في الدرجة الاقتصادية.

- **مطابقة البارامترات:**

يمكن أن تتم المطابقة على مستوى البارامترات بين الخدمة موضوع البحث والخدمات الموجودة في مستودع الخدمات حيث تتم مطابقة بارامترات الدخل وبارامترات الخرج ويتم تحديد درجة المطابقة بالشكل التالي:

1- التطابق التام: في حال التطابق التام بين البارامترين، أو في حال كان أحد البارامترين هو صف جزئي (subclass) من البارامتر الثاني. ففي الأنطولوجية التالية تقع حالة التطابق عندما يقوم مزوّد الخدمة بتقديم خدمة السيارات سواءً كان طلب طالب الخدمة للسيارات أو لسيارات السيدان التي تعتبر فئة من السيارات ضمن الأنطولوجية الموضّحة بالشكل "5-14":



الشكل 5-14، جزء من أنطولوجية المركبات

2- "Plug in": عندما يمثل البارامتر الأول مجموعة تحتوي على البارامتر الثاني.
3- "Failure": عندما لا تكون هناك أية علاقة بين البارامترين.

من الجدير بالذكر أنه من الممكن الدمج بين أكثر من مقاربة للحصول على طريقة مطابقة جديدة بحسب الحاجة. كما ويمكن أن تكون عملية الإستكشاف أكثر تطوّراً وتعقيداً بحيث ترتكز على استنتاج معارف وعلاقات جديدة كلياً انطلاقاً من الحقائق والعلاقات الحالية التي تُعرِّف كل من الخدمة ونموذج تفويض الطلب، وبالطبع فإن هذا يعتمد وبشكل أساسي على درجة تطوّر وتعقيد الأنطولوجيات المُستَخدمة.

4-5. النموذج الرياضي:

فيما يلي مجموعة من التعاريف التي تُوصِّف النموذج S_3OS رياضياً[*1].

1. فضاء العمل:

يتم تعريف فضاء العمل بالسباعية التالية:

W (SERVICES, COMPOSITE_SERVICES, SERVICE_CATALOGUES, ONTOLOGY, Requires, Matches, Imports)

2. تعريف مجموعة الخدمات (SERVICES):

يتم تعريف مجموعة الخدمات الأساسية بالشكل التالي:

SERVICES = {service(prov, prof, mod, gro, reqt): prov ∈ PROVIDERS, prof ∈ PROFILES, mod ∈ MODELS, gro ∈ GROUNDINGS, reqt ∈ REQUEST_DELEGATIONS}

مثال: {خدمة حالة الطقس(الوكالة العالمية الفلكية، ملف الخدمة، نموذج الخدمة، تنفيذ الخدمة، نموذج تفويض الطلب)، خدمة الحجز الفندقي(الفندق الملكي، ملف الخدمة، نموذج الخدمة، تنفيذ الخدمة، نموذج تفويض الطلب)، خدمة حجز رحلات الطيران(الخطوط الجوية السورية، ملف الخدمة، نموذج الخدمة، تنفيذ الخدمة، نموذج تفويض الطلب)،...}

*1 سنتّبع في تسميتنا الطريقة التالية:
أول أحرف العلاقات والتوابع كبير Relation.
أسماء المجموعات بالأحرف الكبيرةSET .
أسماء المتغيرات والعناصر بالأحرف الصغيرة variable.

3. مزوّدي الخدمات (PROVIDERS):

الجهات المزوّدة للخدمات، ومن الممكن أن تكون أية جهة مثل:

PROVIDERS = {Company, Process Engineer..}

4. ملفات الخدمات (PROFILES):

وهي مجموعة تُستَخدم عناصرها لتوصيف الهدف من أحد الخدمات المُضافة عموماً والتي تُستَخدم أثناء تحديد الخدمة باستخدام وصفها الدلالي، وبنفس الطريقة يُستَخدم لتوصيف طلب الخدمة.

PROFILES= {profile (INPUT, OUTPUT,STATEMENT): INPUT, OUTPUT $\in$ φ(PARAMETERS), STATEMENT $\in$ φ(STATEMENTS)}

نقصد من استخدام الرمز φ(SET) التعبير عن مجموعة المجموعات الممكنة التي تنتمي عناصرها إلى هذه الـSET.

مثال:
ملف خدمة التخطيط للرحلات (**{**تاريخ الرحلة، من، إلى، درجة الخدمة في الفندق**}**،
{رقم الرحلة، إسم الشركة، عنوان الفندق، رقم الغرفة**}**،
{خدمة تخطيط الرحلات، من أي مكان في العالم، إلى أي مكان في العالم، حجوزات بدرجات خدمة متنوعة**}**)

5. نماذج الخدمات (MODELS):

وهي مجموعة يُمثّل كل عنصر فيها توصيف الكيفيّة التي تقوم بها أحد الخدمات بأداء الوظيفة المطلوبة منها.

MODELS = {model (process): process $\in$ PROCESSES}

6. ملفات تنفيذ الخدمات (GROUNDINGS):

وهي مجموعة تُمثّل عناصرها توصيف لكيفيّة استخدام أحد الخدمات، وكيف يمكن لطالب الخدمة إستدعاء الخدمة بشكل مباشر.

GROUNDINGS = {grounding (process, wsdl): process $\in$ PROCESSES, WSDL is web service technical invocation description}

إن ملف الـwsdl هو توصيف بلغة الـXML يُمثِّل من الناحية التقنية العقد المباشر بين مزوّد ومستهلك الخدمة، وقد تطرّقنا إليه بالتفصيل في الفصل الثاني. نورد فيما يلي مثال عن هذا الملف الخاص بخدمة الوب الخاصة بالحجز الفندقي:

```
  <?xml version="1.0" encoding="utf-8" ?>
 <wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://tempuri.org/" xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://tempuri.org/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
 <wsdl:types>
<s:schema elementFormDefault="qualified"
targetNamespace="http://tempuri.org/">
<s:element name="HelloWorld">
   <s:element name="HelloWorldResponse">
<s:element name="NewReservation">
<s:complexType>
<s:sequence>
<s:element minOccurs="0"maxOccurs="1" name="VisitorName" type="s:string"/>
<s:element minOccurs="1" maxOccurs="1"name="FromDate"type="s:dateTime" />
  <s:element minOccurs="1" maxOccurs="1" name="ToDate" type="s:dateTime" />
  <s:element minOccurs="1" maxOccurs="1" name="RoomID" type="s:int" />
  <s:element minOccurs="1" maxOccurs="1" name="RowID" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="HotelName" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="HotelPhone" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="HotelURI" type="s:string" />
  </s:sequence>
  </s:complexType>
  </s:element>
<s:element name="NewReservationResponse">
  </s:schema>
  </wsdl:types>
<wsdl:message name="HelloWorldSoapIn">
<wsdl:message name="HelloWorldSoapOut">
<wsdl:message name="NewReservationSoapIn">
<wsdl:message name="NewReservationSoapOut">
 <wsdl:portType name="ServiceSoap">
<wsdl:binding name="ServiceSoap" type="tns:ServiceSoap">
 <wsdl:binding name="ServiceSoap12" type="tns:ServiceSoap">
<wsdl:service name="Service">
  </wsdl:definitions>
```

7. نموذج تفويض الطلب (REQUET_DELEGATIONS):

وتحتوي مجموعة من العناصر يُمثّل كل منها توصيف طالب الخدمة للإمكانيات الوظيفية المطلوبة، والتي يجب أن تحتوي على وصف إمكانيات خدمة الوب المطلوبة.

REQUEST_ DELEGATIONS = {request_delegation (STATEMENT): STATEMENT ∈ φ(STATEMENTS)}

مثال نموذج تفويض الطلب لخدمة تخطيط الرحلات:
نموذج تفويض الطلب (**{**تخطيط الرحلات في الدول الأوروبية، إعداد جولات سياحية برية وبحرية، حجوزات طيران بدرجات متنوعة، حجوزات فندقية بدرجات متنوعة**}**).

8. مجموعة المهام (PROCESSES):

وهي مجموعة المهام التي تُمثِّل الإجرائيات المقابلة للخدمات.

PROCESSES = {process (name, INPUT, OUTPUT): INPUT, OUTPUT ∈ φ(PARAMETERS), name ∈ PRIMITIVES}

مثال عملية حجز بالطائرة:
العملية (Reserve، **{**من، إلى، اليوم**}**، **{**رقم الرحلة، اسم الشركة**}**).

9. مجموعة التعابير الوصفية (STATEMENTS):

وهي مجموعة التعابير الوصفيّة المُستَخدمة لتوصيف خدمة ما في فضاء الخدمات. يمكن توليد هذه العبارات من خلال العلاقة التالية:

CONCEPTS = {c: c is a concept}
RELATIONS = {r: r is a relation between two concepts}
Statement: CONCEPTS X RELATIONS → CONCEPTS

الـConcept هو عبارة عن مفهوم ما مثل خدمة، رحلة، فندق...

الـRelation وهي عبارة عن علاقة تربط بين المفاهيم مثل علاقة "هو أو هي" من مثل الفندق هو مكان إقامة.

10. مجموعة المعاملات (PARAMETERS):

وهي معاملات خدمات الوب، وقد تكون معاملات دخل تتطلبها الخدمة للتمكن من تنفيذ وظيفتها، وقد تكون معاملات خرج تَنتج عن تنفيذ خدمة الوب.

PARAMETERS= {parameter (name, type, datatype): name, type, datatype ∈ PRIMITIVES}

مثال: {(تاريخ المغادرة، معامل دخل، من النوع تاريخ)،(مطار الوصول، معامل دخل، من النوع مطار)، (رقم الحجز، معامل خرج، من النوع رقم) ...}

11. PRIMITIVES:

وهي مجموعة المواصفات التي يتم استخدامها لتوصيف أو تنفيذ الخدمات، يتم تزويدها من قبل مزوّد الخدمة أو من قبل طالب الخدمة تبعاً لنوع العملية.

PRIMITIVES= {نص, رقم, دخل, خرج,...}

12. تعريف الخدمة المركّبة:

وتُمثل مجموعة من الخدمات المرتبطة ببعضها لتوليد وظيفة أو خدمة جديدة، ويتم توصيف الهدف منها بشكل دلالي وكيفيّة تحقيق هذه الوظيفة.

COMPOSITE_SERVICES ={ composite_service(THESERVICE, prov ,prof, mod, reqt): THESERVICE ∈ φ(THE_SERVICES), prof ∈ PROFILES, mod ∈ COMPOSITE_ MODELS, reqt ∈ REQUEST_DELEGATIONS)}

مثال: {خدمة التخطيط للرحلات(**{**خدمة حجز رحلات الطيران، خدمة الحجز الفندقي**}**، مكتب دمشق السياحي، ملف الخدمة، نموذج الخدمة المركّبة، نموذج تفويض الطلب)، خدمة الشراء الإلكتروني(**{**خدمة شراء المواد، خدمة شحن البضائع**}**، مكتب حمص التجاري الإقليمي، ملف الخدمة، نموذج الخدمة المركّبة، نموذج تفويض الطلب)،...}

13. نماذج الخدمات المركّبة (COMPOSITE_MODULES):

وهي مجموعة يُمثّل كل عنصر فيها توصيف الكيفيّة التي تقوم من خلالها إحدى الخدمات بأداء الوظيفة المطلوبة منها.

COMPOSITE_MODELS= {composite_model (PROCESS, REQUEST_ DELEGATION): PROCESS ∈ φ(PROCESSES), REQUEST_DELEGATION ∈ φ(REQUEST_ DELEGATIONS)}

14.مجموعة الخدمات (THE_SERVICES):

وهي مجموعة يُمكن الحصول عليها من خلال اجتماع كل من مجموعتي الخدمات والخدمات المركّبة بالشكل التالي:

THE_SERVICES=SERVICES $\cup$ COMPOSITE_SERVICES

مثال:

THE_SERVICES= {خدمة الحجز الفندقي، خدمة حجز رحلة طيران، خدمة شحن البضائع،...} $\cup$ {خدمة التخطيط لرحلة،خدمة الشراء الإلكتروني،...}

15.العلاقة Requires:

وتُعبِّر عن ارتباط الخدمة المركّبة بمجموعة من الخدمات التي تتطلب تنفيذها.

Requires: COMPOSITE_SERVICES $\rightarrow \varphi$(THE_SERVICES) Requires(cs) =Y: $\forall$cs $\in$ COMPOSITE_SERVICES,$\exists$ Y $\in$ φ(THE_SERVICES), cs $\notin$ Y

مثال:

Requires (خدمة التخطيط للرحلات)= {خدمة حجز رحلة الطيران، خدمة الحجز الفندقي}

نستنتج من هذه العلاقة مايلي:

نتيجة 1:

Requires (cs1) =X, Requires(x) =Z $\Longrightarrow$ Requires (cs1) =(X\{x}) $\cup$ Z, x $\in$ X

مثال:

Requires (خدمة التخطيط للرحلات) = { خدمة حجز رحلة الطيران، خدمة الحجز الفندقي، خدمة تخطيط الجولات الميدانية السياحية}

Requires (خدمة تخطيط الجولات الميدانية السياحية) = { خدمة حجز الجولات السياحية البرية، خدمة حجز الجولات السياحية البحرية}

$\Longrightarrow$ Requires (خدمة التخطيط للرحلات) = {خدمة حجز رحلة الطيران، خدمة الحجز الفندقي، خدمة حجز الجولات السياحية البرية، خدمة حجز الجولات السياحية البحرية}

نتيجة 2:

نقول عن خدمة ضمن مجموعة الخدمات على أنها مركّبة إذا كانت تتطلب تنفيذ خدمات أخرى لتتمكن من أداء وظيفتها:

s ∈ THE_SERVICES, Requires(s) ≠ ∅ ⟹ s is composite service

نتيجة 3:

نقول عن خدمة ضمن مجموعة الخدمات على أنها بسيطة إذا كانت لا تتطلب تنفيذ أية خدمة أخرى:

s ∈ THE_SERVICES, Requires(s) =∅ ⟹ s is simple service

16.ملف قائمة الخدمة (SERVICE CATALOGUES):

وهو ملف سيناريوهات تَكَامُل الخدمات مع خدمات أخرى.

SERVICE_CATALOGUES = {service_catalogue(s, Y): s ∈ THE_SERVICES, Y ∈ φ(cs), cs ∈ COMPOSITE_SERVICES, s ∈ Requires(cs)}

مثال: ملف قائمة خدمة الحجز الفندقي قد تتألف من مجموعة الخدمات التالية: خدمة تخطيط الرحلات حول العالم، خدمة تخطيط الرحلات في الشرق الأوسط، خدمة الحجوزات الفندقية.

17.علاقة المطابقة (Matches):

وتُحدِد هذه العلاقة مجموعة الخدمات المطابقة لخدمة ما.

Matches: THE_SERVICES → φ(THE_SERVICES) Matches(x)=Y: x ∈ THE_SERVICES, ∀ y ∈ Y, Matched(x,y)=1

18.العلاقة Matched:

الشرط اللازم لتطابُق خدمتين هو تطابُق معاملاتها وتطابُق عباراتها الوصفية.

Matched: THE_SERVICES X THE_SERVICES → {0, 1}

Matched (x1, x2) =
- 1: x1, x2 ∈ THE_SERVICES, ∀ a ∈ Present(x1), ∃ b ∈ Present(x2), a=b
- 0: x1, x2 ∈ THE_SERVICES, ∀ a ∈ Present(x1), ∄ b ∈ Present(x2), a=b

19. العلاقة Presents:

وتُمثّل العلاقة بين الخدمة وملف الخدمة بحيث ترتبط كل خدمة بملف خدمة وحيد.

Presents: THE_SERVICES → PROFILES
Presents(x)=z: Presents is a bijection, if Presents(x)=Presents(y) then x=y

20. العلاقة DescribedBy:

وتمثّل العلاقة بين الخدمة ونموذج الخدمة بحيث ترتبط كل خدمة بنموذج خدمة وحيد.

DescribedBy: THE_SERVICES → MODELS
DescribedBy(x)=z: DescribedBy is a bijection, if DescribedBy(x)= DescribedBy(y) then x=y

21. تعريف مكامل الخدمات:

المهمة الأساسية لمكامل الخدمات تتمثل بالعلاقة Generate التي تربط الخدمات ببعضها لتوليد خدمات مركّبة جديدة.

SI (SERVICES, COMPOSITE_SERVICES, Generate)
Gen: SERVICES X COMPOSITE_SERVICES → COMPOSITE_SERVICES
Gen(x,y)=z: x ∈ SERVICES, y, z ∈ COMPOSITE_SERVICES

5-5. خلاصة:

قمنا في هذا الفصل باستخدام الأنطولوجية الخاصة بأحد لغات الأنطولوجية المُستَخدمة في خدمات الوب الدلالي OWL-S وتطويرها بحيث نحصل على أنطولوجية يمكن من خلالها تمثيل الخدمة المُستَهلكة نفسها كطالب لخدمة أخرى مُكافِئة لها، إلى جانب ما يتطلبه ذلك من نشر الخدمات واستكشافها ومكاملتها من خلال عرض النموذج S_3OS نظرياً ورياضياً.

النموذج العملياتي

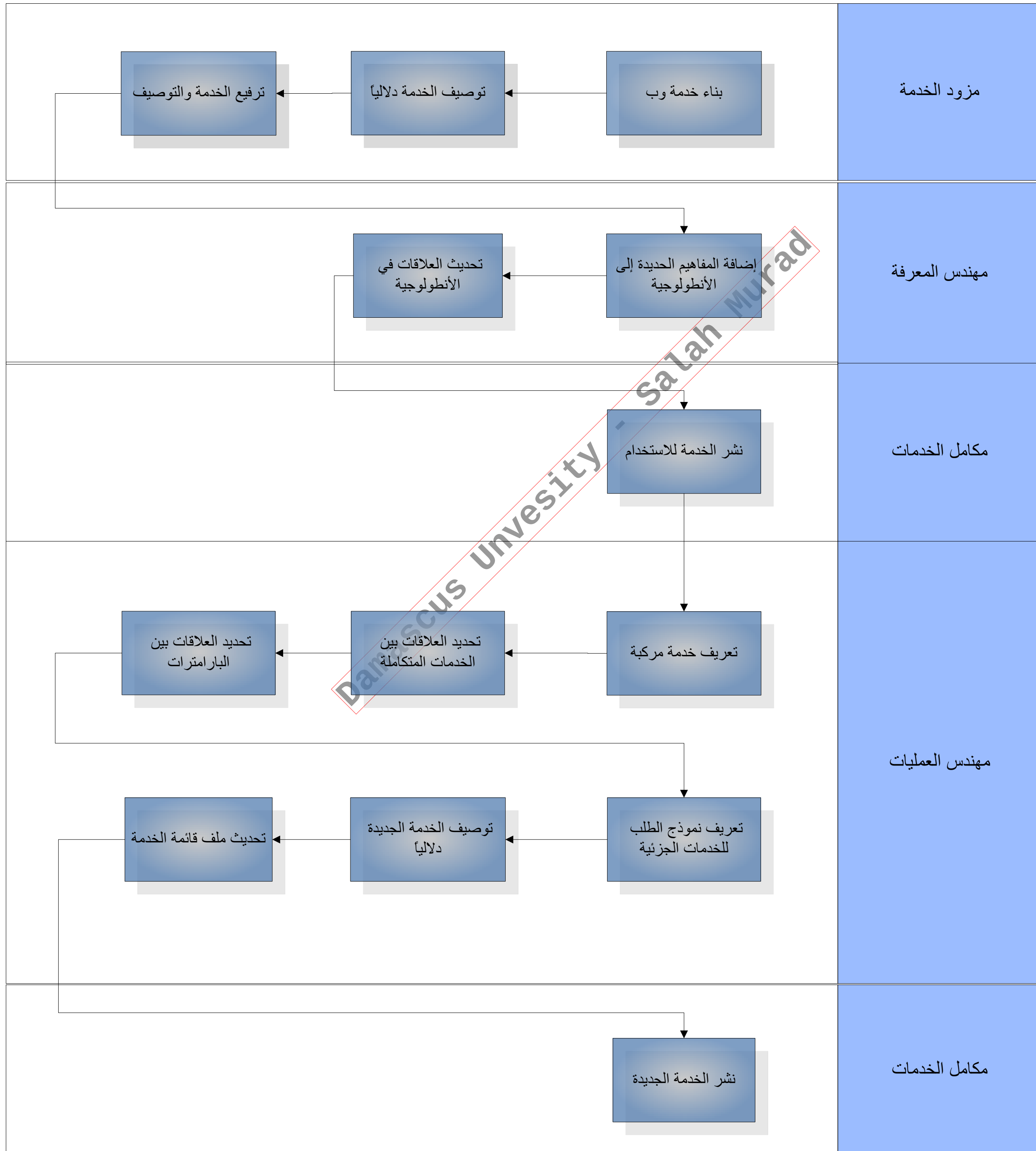
Operation Model

مقدّمة:

بعد استعراض النموذجين النظري والرياضي لبناء النظم خدمية التوجّه باستخدام خدمات الوب المُوصّفة دلالياً، سنقوم بطرح آلية العمل الضرورية لبناء هذه الأنظمة من وجهة نظر كلٍ من مزوّد الخدمة عند نشر الخدمة ومن وجهة نظر مستخدم الخدمة عند استهلاكها، كما سنقوم بتحديد علاقة المكوّنات الأساسية ضمن إطار عمل مكامل الخدمات ومجموعة الخوارزميات التي يحتاجها.

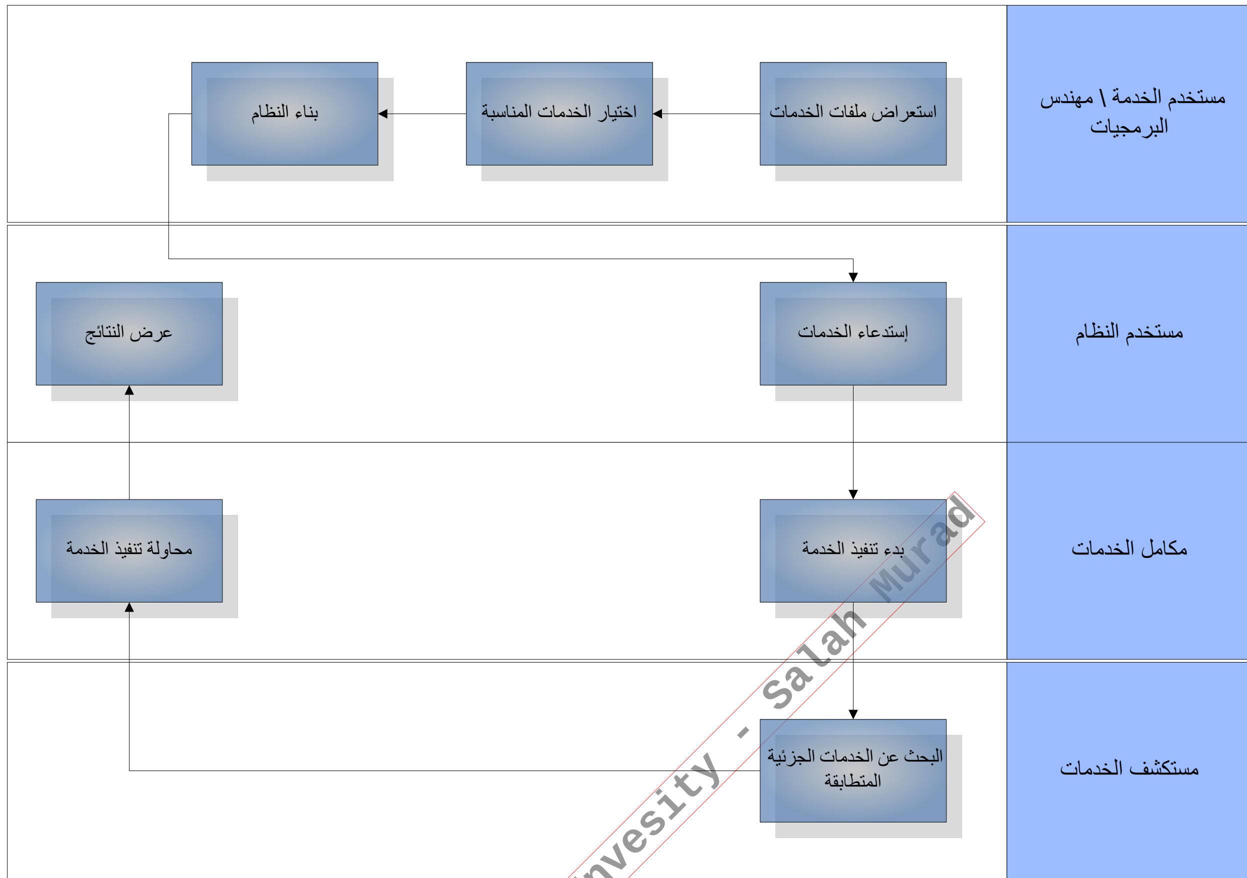
1-6. آلية العمل باستخدام نموذج الـS_3OS:

قبل الخوض في خوارزميات النموذج العملياتي سنقوم بطرح آلية العمل (Process) التي نراها مناسبة ليتم من خلالها الوصول إلى الأهداف المرجوّة من استخدام البعد الدلالي في خدمات الوب المُستخدَمة لبناء المنظومات خدمية التوجه وذلك من عدة وجهات نظر. حيث يعرض الشكل "1-6" الأدوار والوظائف التي يجب القيام بها لدى القيام بتزويد أية خدمة جديدة.



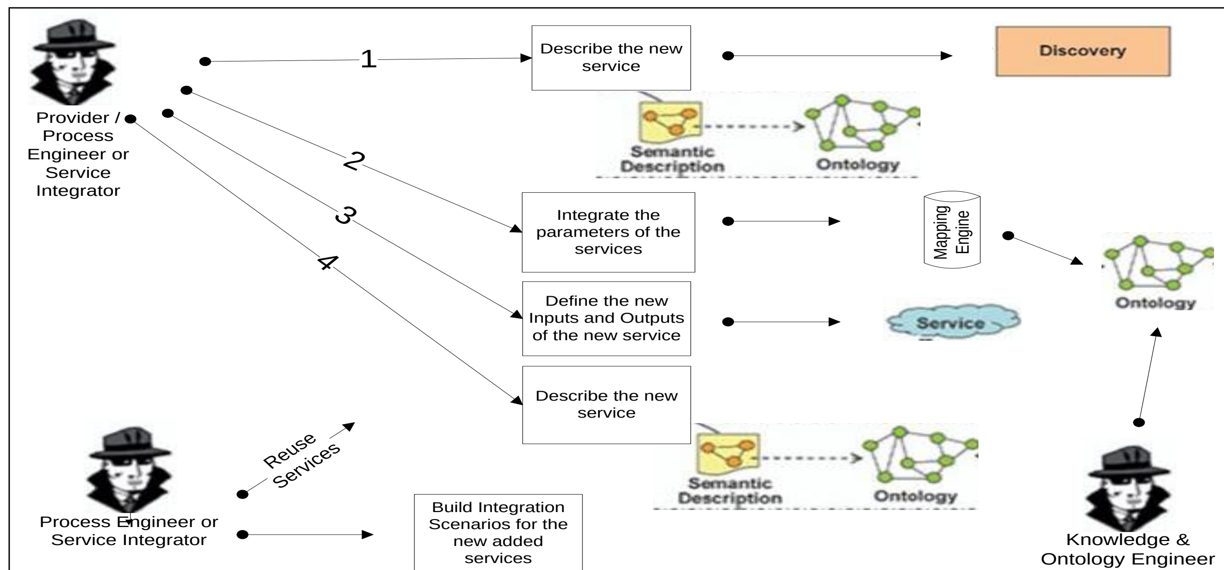
الشكل 6-1، إضافة خدمة جديدة

بينما يوضح الشكل "6-2" العمليات والمهام المطلوبة عند استخدام الخدمات.



الشكل 6-2، استخدام الخدمات عن طريق مكامل الخدمات

يمكن تلخيص أهم العمليات الأساسية التي تم ذكرها في الشكلين السابقين (6-1، 6-2) من خلال الشكل التالي "6-3".



الشكل 6-3، العمليات الأساسية لإضافة واستخدام الخدمات عن طريق مكامل الخدمات

يوضّح المخطط السابق (6-3) آلية عمل يتم من خلالها إيضاح دور مهندس العمليات أومكامل الخدمات في عملية بناء وتوصيف الخدمات المركبة أوفي بناء سيناريوهات مكاملة إفتراضية بالإعتماد على الخدمات البسيطة الجديدة المُضَافة إلى مخزن الخدمات أو بالإعتماد على خدمات مركّبة أخرى بشكل مشابه لدور مهندس المعرفة أو مدير الأنطولوجية عند إضافة مفاهيم جديدة إلى قاعدة البيانات، ويتم تحقيق هذا من خلال مكوّن جديد لنماذج الخدمات الدلالية وهو مكامل الخدمات.

يتم من خلال مخطط العمل هذا التركيز على بناء خدمات جديدة وليس مجرد توصيف الخدمات مما يتيح إمكانية إعادة استخدام الخدمات مما يساعد في إتاحة إمكانيات أوسع لبناء وانتشار أنظمة العمل المعتمدة على الخدمات. إن مدى كفاءة هذا يعتمد على مدى كفاءة وديناميكية مكامل الخدمات وما يتمتع به من مواصفات لبناء الخدمات وسيناريوهات المكاملة.

إن الحديث عن مكاملة مجموعة من الخدمات والتي يتم تقديمها من قبل أكثر من مزوّد خدمة يعني بالضرورة إمكانية وجود إختلاف في المفاهيم المُستَخدمة في هذه الخدمات وتوصيفها، وهنا يأتي دور الأنطولوجية المُمَثَّلة في المخطط السابق لفض النزاعات بين المفاهيم، وكلما زاد حجم العمل نكون بحاجة لأنطولوجية أشمل وأكبر حجماً بحيث تكون أكثر قدرة وفاعلية على أداء دور فعال للتعامل مع مفاهيم متعددة من قبل مزوّدي خدمات متعددين، وهذا ما يُضفي بعداً دلالياً لعملية المكاملة بين الخدمات.

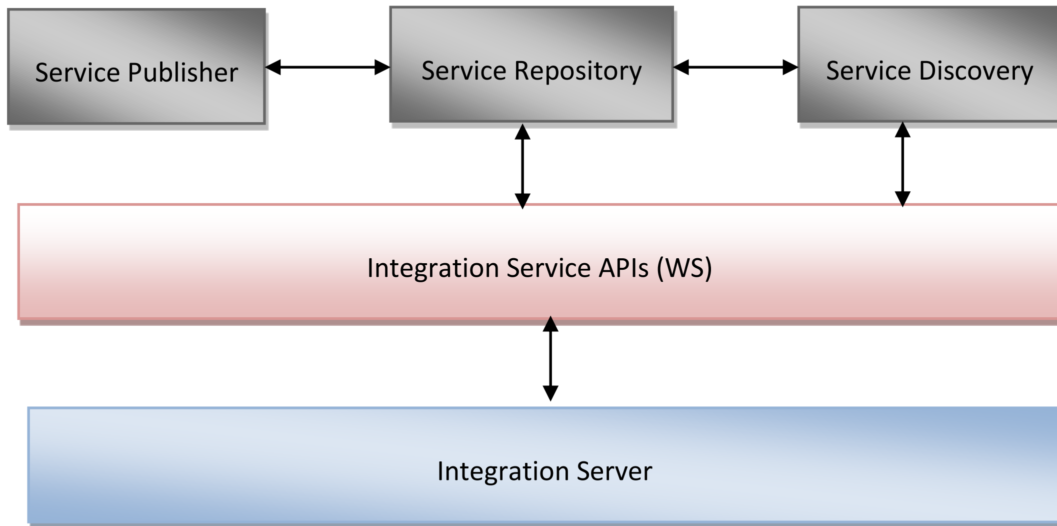
عند إضافة خدمة جديدة موصفة دلالياً من قبل أحد مزوّدي الخدمة يقوم مدير الأنطولوجية بإضافة المفاهيم الجديدة المُستَخدمة، ويقوم مهندس العمليات أو مزوّد الخدمة بتعريف آلية مكاملة هذه الخدمة بخدمات أخرى موجودة بشكلٍ مسبق أو إضافة سيناريوهات متعدّدة للمكاملة، ويقوم بتوصيف الخدمات الجديدة التي تم توليدها بشكل دلالي أيضاً ليسهل إيجادها والتعامل معها بنفس الطريقة.

سيتيح مكامل الخدمات إمكانية المكاملة بين الخدمات بالإعتماد على وصفها الدلالي وليس على إسمها أوعنوانها فقط، مما يعني فصل الخدمة عن مزوّديها مما يتيح استخدام أية خدمة تمتلك نفس الوصف الدلالي بغض النظر عن مزوّدها وعنوانها، وهذ كفيل بإعطاء بعد ديناميكي وأكثر مرونة لعملية بناء واستخدام الخدمات بحيث يتيح إمكانية استخدام بدائل لأية خدمة لا يمكن استخدامها حالياً لسببٍ ما، كأن تكون غير متاحة حالياً، كما ستتيح هذه الطريقة إعادة الإستخدام بأقل تدخّل ممكن وبشكل أكثر ديناميكية.

6-2. إطار العمل الخاص بمكامل الخدمات:

يُقدِّم مخدّم مكامل الخدمات الوظائف التي يتيحها في مجال ربط ومكاملة الخدمات في إطار العمل الذي نُقدّمه (Framework) من خلال مجموعة من التوابع كطبقة تقوم بتغلف المهام الرئيسية التي تقوم بها المكونات الأساسية في إطار العمل الموضّح من خلال الشكل "6-4" وهي: محرك إستكشاف الخدمات، وناشر الخدمات، ومستودع الخدمات، ومخدّم المكاملة.

يتيح إطار العمل لمزوّدي الخدمات نشر الخدمات التي يقومون بتأمينها من خلال ناشر الخدمات، حيث يقوم مزوّد الخدمة بترفيع معلومات وتوصيف الخدمة وعنوانها، تتم إضافة هذه الخدمة إلى مستودع الخدمات، هذا ويتم البحث عن الخدمات ومطابقتها من خلال مستكشف الخدمات. يقوم مخدّم مكاملة الخدمات بالتفاعل مع كل من المكونات السابقة وذلك من خلال مجموعة من التوابع.



الشكل 6-4، إطار عمل مكامل الخدمات

3-6. بنية التطبيقات في نموذج نظم المعلومات خدمية التوجه الدلالي:

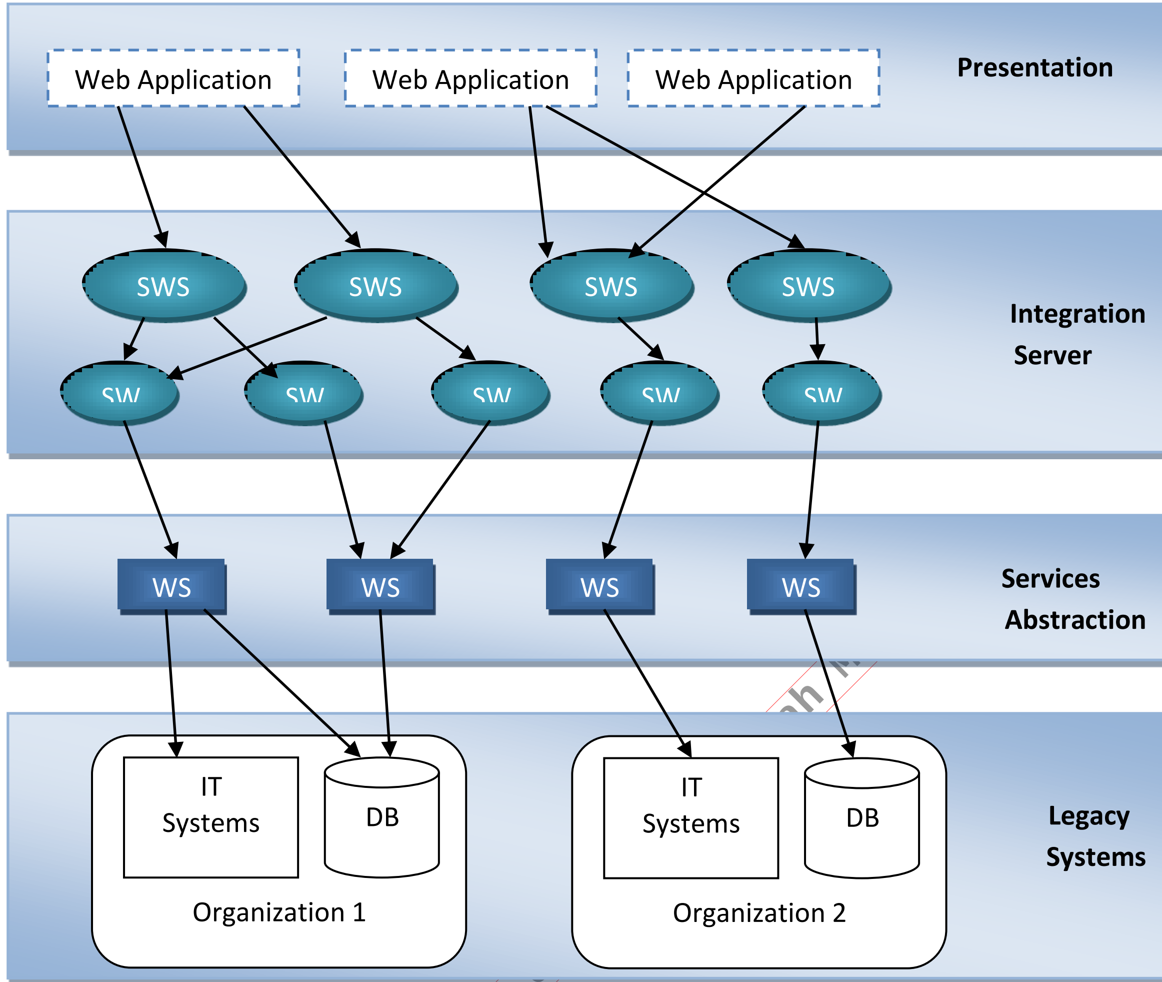
إن البنية العامة للتطبيقات المعتمدة على الخدمات البسيطة أو المركّبة والموصّفة دلالياً التي نُقدِّمها تتألف من مجموعة من الطبقات كما هو موضّح في الشكل "6-5". حيث تتيح هذه البنية إمكانية عرض الوظائف التي تقدمها الأنظمة الموجودة بشكلٍ مُسبق كخدمات وب يمكن إستخدامها من قبل أنظمة أخرى، ثم يتم توصيفها بشكل دلالي وترفيعها إلى مستودع الخدمات الموصّفة دلالياً مع إمكانية مكاملتها مع خدمات أخرى وذلك باستخدام مكامل الخدمات. هذه الطبقات هي:

1- طبقة الأنظمة:
وتتألف هذه الطبقة من أنظمة المعلومات ومصادر البيانات الخاصة بالجهات المُستفيدة من عملية المكاملة بين التطبيقات.

2- طبقة تجريد الخدمة:
وفيها يتم عرض الوظائف الجزئية (micro functionality) التي تُقدِّمها أنظمة المعلومات المُمَثلة في الطبقة السابقة بشكلٍ مجردٍ عن مواصفات العتاد أو بيئة العمل التي تعمل من خلالها.

3- طبقة خدمات الوب الدلالية:
تتوضّع في هذه الطبقة مجموعة الخدمات الموصّفة بشكل دلالي، والتي يتم استكشافها والوصول إليها من خلال استخدام نموذج تفويض الطلب لطلب خدمة ما، حيث يقوم مخدّم مكامل الخدمات بإجراء عملية الاستكشاف والمطابقة واستدعاء مجموعة الخدمات المناسبة بالإعتماد على الوصف الدلالي والأنطولوجية المُستَخدمة للوصول إلى الوظيفة المطلوبة.

4- طبقة العرض:
وتُمثل هذه الطبقة مجموعة تطبيقات الوب أو أنظمة المعلومات التي تستخدم الوب والتي تعتمد في بنائها على خدمات الوب الموصفة دلالياً. حيث يمكن وجود أكثر من تطبيق وب يستخدم مجموعة معينة من خدمات الوب لتغطية احتياجات فئة معينة من زبائنه والتي قد تختلف عن أهداف واحتياجات فئة أخرى.



الشكل 6-5، بنية تطبيقات خدمات الوب الدلالي

من أجل الوصول بنجاح إلى بناء التطبيقات بالاعتماد على خدمات الوب الدلالية بشكل عام بما يتناسب مع البنية التي تم وصفها سابقاً، يجب مراعاة الخطوات التالية:

1- تحديد المتطلبات:
حيث يتم تحديد متطلبات التطبيق باستخدام أحد منهجيات هندسة البرمجيات المعيارية، بحيث يتم تحديد المتأثرين بالتطبيق (*Stakeholders*)، المستخدمين والأدوار، المزوّدين المحتملين لخدمات الوب، وأية متطلبات أخرى قد تتعلق بالبنية التحتية (Infrastructure) والواجهات (Interfaces).

2- توصيف الهدف:
بالاعتماد على وثيقة المتطلبات يتم تحديد الأهداف التي سيتم إستخدامها ويتم توصيفها في مخدّم مكاملة الخدمات، وأثناء هذه العملية يتم استخدام المفاهيم الموجودة في الأنطولوجية المُستَخدمة وعند الحاجة تتم إضافة المفاهيم الجديدة.

3- توصيف خدمات الوب:
يتم توصيف خدمات الوب باستخدام الأنطولوجية، وأيضاً مع إمكانية إضافة المفاهيم الجديدة.

إن تحقيق الخطوات السابقة يتطلب تدخل عدة أطراف تبدأ من الزبون أوالمستخدم، والمتأثرين بالخدمات أو التطبيقات، ومطوّري خدمات الوب الدلالية، ومهندس المعرفة، ومهندس العمليات.

4-6. النموذج العملياتي:

فيمايلي عرض لمجموعة من أهم خوارزميات العمل.

1. تحديث الأنطولوجية:

الدخل: الأنطولوجية المُستَهدفة، ومجموعة المفاهيم الجديدة، ومجموعة العلاقات بين هذه المفاهيم.

الخرج: الأنطولوجية المُعدَّلة.

الخوارزمية:

```
//O typeof Ontology
//Cs listofType Concept
//Rs listofType Relation

refreshOntology(O،Cs،Rs)
if(O is exist)
     foreach (c in Cs)
          addConcept(O،c)
     foreach (r in Rs)
          addRelation(O،r،r.concept1،r.concept2)
     return O
else
     return Null
```


2. إضافة مفهوم جديد:

الدخل: الأنطولوجية المُستَهدفة، والمفهوم الجديد.

الخرج: الأنطولوجية المُعدَّلة.

الخوارزمية:

```
//O typeof Ontology
//C typeof Concept

addConcept(O،C)
if(O is exist)
    if Not findConcept(O،C)
      Concepts ← Concepts ∪{C} where Ontology ≡ O
     return O
else
     return Null
```

3. إضافة علاقة جديدة:

الدخل: الأنطولوجية المُستَهدفة، والعلاقة الجديدة، والمفهوم الأول، والمفهوم الثاني للعلاقة.

الخرج: الأنطولوجية المُعدَّلة.

الخوارزمية:

```
//O typeof Ontology
//R  typeof Relation
//Concept1  typeof Concept
//Concept2  typeof Concept

addRelation(O،R،Concept1،Concept2)
if(O is exist)
 if findConcept(O،Concept1)
  if  findConcept(O،Concept2)
    if Not findRelation(O،R،Concept1،Concept2)
      Relations ←Relations ∪{R}where Ontology ≡ O
     return O
  else
    addConcept(O،Concept2)
    addRelation(O،R،Concept1،Concept2)
else
    addConcept(O،Concept1)
    addRelation(O،R،Concept1،Concept2)
else
     return Null
```


4. البحث عن مفهوم ضمن أنطولوجية:

الدخل: الأنطولوجية المُستَهدفة، والمفهوم المُراد البحث عنه.

الخرج: نتيجة بوليانية تدلّ على نجاح العملية.

الخوارزمية:

```
//O typeof Ontology
//C  typeof Concept

findConcept(O،C)
if(O is exist)
    foreach con in O.Concepts
        if con=C
            return true
    return false
else
    return false
```

5. إضافة توصيف الخدمة:
الدخل: الخدمة المُوصَّفة، والأنطولوجية المُستَخدمة، ومجموعة التعابير التوصيفية.
الخرج: نتيجة بوليانية تدلّ على نجاح العملية.
الخوارزمية:

```
//O typeof Ontology
//S typeof Service
//Statements listofType Statement

addServiceDescription(O،S،Statements)
if(O is exist)
  if findService(S)
   foreach s in Statements
    p=getProfile(S)
    if Not findStatement(p،s)
            addStatement(p،s)
   return true
  else
     return false
else
  return false
```

6. تعريف بارامترات خدمة بسيطة:
الدخل: الخدمة، والأنطولوجية المُستَخدمة، ومجموعة بارامترات الخدمة ونوعها وطبيعتها.
الخرج: نتيجة بوليانية تدلّ على نجاح العملية.
الخوارزمية:


```
//O typeof Ontology
//S typeof Service
//Parameters listofType Parameter

addServiceParameters(O،S،Parameters)
if(O is exist)
  if findService(S)
   foreach para in Parameters
    p=getProfile(S)
    if Not findParameter(p،para)
            addParameter(p،para.Name،
                           para.Type،
                           para.DataType)
   return true
  else
     return false
else
  return false
```

7. تعريف بارامترات خدمة مركّبة:
الدخل: الخدمة، والأنطولوجية المُستَخدمة، ومجموعة بارامترات الخدمة المركبة ونوعها وطبيعتها وعلاقتها ببارامترات الخدمات الأخرى.
الخرج: نتيجة بوليانية تدلّ على نجاح العملية.
الخوارزمية:

```
//O typeof Ontology
//CS typeof CompositeService
//Parameters listofType Parameter

addCompositeServiceParameters(O،CS،Parameters)
if(O is exist)
  if findService(CS)
   foreach para in Parameters
    p=getProfile(CS)
    if Not findParameter(p،para)
            addParameter(p،para.Name،
                        para.Type،
                        para.DataType،
                        para.DestinationParameter)
   return true
  else
     return false
else
  return false
```


8. البحث عن خدمة باستخدام نموذج تفويض الطلب:
الدخل: نموذج تفويض الطلب.
الخرج: الخدمات المُطابِقة لنموذج تفويض الهدف.
الخوارزمية:

```
//RT typeof RequestDelegation

findServiceByRequestDelegation(RT)
   foreach service in Services
    if Match(service.Profile،RT)
     matchedServices ←matchedServices ∪ {service}
    return matchedServices
```

9. تحديث بارامترات الخدمات في خدمة مركّبة:
الدخل: الخدمة المركّبة.
الخرج: نتيجة بوليانية تدلّ على نجاح العملية.
الخوارزمية:

```
//CS typeof Service

updateServiceParameters(CS)
   foreach service in CS.Services
        foreach para in service.parameters
            para.setValue=
                para.SourceParameter.getValue
        if service is composite
            updateServiceParameters(service)
   return true
```

10. تكامل بارامترات الخدمات في خدمة مركّبة:

الدخل: الخدمة المركبة، وقائمة بالبارامترات وعلاقاتها.

الخرج: نتيجة بوليانية تدلّ على نجاح العملية.

الخوارزمية:

```
//CS typeof Service
//Parameters listofType Parameter

integrateServiceParameters(CS،Parameters)
   foreach service in CS.Services
        foreach para in service.parameters
            para1=getParameter(Parameters،para)
            para.SourceParameter=
                        para1.SourceParameter
            refreshParameters(para،Parameters)
        if service is composite
            integrateServiceParameters(
                            service،
                            Parameters)
   return true
```

11. إرجاع ملف خدمة:

الدخل: الخدمة المُستَهدفة.

الخرج: ملف الخدمة.

الخوارزمية:

```
//CS typeof Service

getProfile(CS)
   foreach service in Services
      if service=CS
        return service.Profile
   return Null
```

12. إضافة مزوّد خدمة:

الدخل: مزوّد خدمة.

الخرج: قائمة مزوّدي الخدمات.

الخوارزمية:

```
//SP typeof ServiceProvider

addServiceProvider(SP)
    if Not findServiceProvider(SP)
        ServiceProviders ←ServiceProviders ∪ {SP}
    return ServiceProviders
```

13. البحث عن عبارة:

الدخل: ملف الخدمة، والعبارة المُراد البحث عنها.

الخرج: نتيجة بوليانية تُعبِّر عن إيجاد العبارة من عدمه.

الخوارزمية:


```
//P typeof Profile
//S typeof Statement

findStatement(P،S)
if P is exist
    foreach statement in P.statements
     if statement.Concept1=S.Concept1 or
        Equvalient(statement.Concept1،S.Concept1)
         if statement.Concept2=S.Concept2 or
         Equvalient(statement.Concept1،S.Concept1)
          if statement.Relation=S.Relation
            return true
  return false
else
    return Null
```

14. إضافة عبارة إلى ملف خدمة:

الدخل: ملف الخدمة، والعبارة المُراد إضافتها.

الخرج: ملف الخدمة المُعدَّل.

الخوارزمية:

```
//P typeof Profile
//S typeof Statement

addStatement(P،S)
```

```
    if P is exist
        if Not findStatement(P،S)
            P.Statements←P.Statements ∪ {S}
    return P
    else
        return Null
```

15. إضافة بارامتر إلى ملف خدمة:

الدخل: ملف الخدمة، والبارامتر المُراد إضافته.

الخرج: البارامتر المُضاف.

الخوارزمية:

```
//P typeof Profile
//name typeof Primitive
//type typeof Primitive
//datatype typeof Primitive

addParameter(P،name،type،datatype)
    if P is exist
        if Not findParameter(P،
                 Parameter(name،type،datatype))
            para= Parameter(name،type،datatype)
            P.Parameters=P.Parameters ∪ {para}
        return  para
    else
        return Null
```


16. إضافة بارامتر إلى ملف خدمة:

الدخل: ملف الخدمة، والبارامتر المُراد إضافته.

الخرج: البارامتر المُضاف.

الخوارزمية:

```
//P typeof Profile
//name typeof Primitive
//type typeof Primitive
//datatype typeof Primitive
//paraDestination typeof Parameter

addParameter(P،name،type،datatype،paraDestination)
    if P is exist
        if findParameter(P،paraDestination)
         para=addParameter(P،name،type،datatype)
          if para is not Null
            paraDestination.SourceParameter =para
        else
```

```
                return false
        return Para
    else
        return Null
```

17. البحث عن بارامتر في ملف خدمة:

الدخل: ملف الخدمة، والبارامتر المُراد البحث عنه.

الخرج: نتيجة بوليانية تدلّ على العثور على البارامتر.

الخوارزمية:

```
//P typeof Profile
//Para typeof Parameter

findParameter(P،Para)
    if P is exist
        foreach para1 in P.Parameters
            if para1.Name=Para.Name and
               para1.Type=Para.Type and
               para1.DataType=Para.DataType
                 return true
     return false
     else
         return Null
```


18. إضافة خدمة:

الدخل: الخدمة وما ترتبط به من توصيف وظيفي، نموذج الخدمة، والتوصيف التقني اللازم للتنفيذ.

الخرج: قائمة كل الخدمات بعد إضافة الخدمة الجديدة.

الخوارزمية:

```
//O typeof Ontology
//S typeof Service
//P typeof Profile
//M typeof Model
//G typeof Grounding

addService(O،S،P،M،G)
    if O is exist
        if (Not FindService(S))
            Services←Services ∪ {S}
            Present(S)=P
            Describe(S)=M
            Support(S)=G
            foreach statement in P.Statements
                if Not findConcept(O،
                           statement.Concept1)
```

```
                    Concepts.add(
                    statement.Concept1)
                if Not findConcept(O،
                       statement.Concept2)
                    Concepts.add(
                    statement.Concept2)
                if Not findRelation(O،
                       statement.Relation)
                    Relations.add(
                    statement.Concept2)
            refreshOntology(O،Concepts،Relations)
            return Services
        else
            return Services
    else
        return Null
```

19. إضافة خدمة مركّبة:

الدخل: الخدمة موضوع المُكاملة، وتوصيف وظيفة الخدمة الجديدة، ونموذج الخدمة المركّبة المضافة.

الخرج: عنوان استدعاء الخدمة الجديدة.

الخوارزمية:


```
//O typeof Ontology
//Sc listofType Service
//P typeof Profile
//M typeof Model
//CS typeof CompositeService

addCompositeService(O،CS،Ss،P،M)
    if O is exist
        foreach service in Ss
            if findService(service)=false
                return false
        ServiceCatalogue(S)←
                  ServiceCatalogue(S) ∪ {CS}
        Services←Services ∪ {CS}
        Present(CS)=P
        Describe(CS)=M
        integrateServiceParameters(
                   CS،CS.Parameters)
        foreach statement in P.Statements
                if Not findConcept(O،
                       statement.Concept1)
                    Concepts.add(
                    statement.Concept1)
                if Not findConcept(O،
```

```
                        statement.Concept2)
                     Concepts.add(
                     statement.Concept2)
                 if Not findRelation(O،
                        statement.Relation)
                     Relations.add(
                     statement.Concept2)
             refreshOntology(O،Concepts،Relations)
         return Services
     else
       return Null
```

20. البحث عن خدمة:

الدخل: الخدمة المُراد البحث عنها.

الخرج: نتيجة البحث المنطقية.

الخوارزمية:

```
//S typeof Service

findService(S)
  foreach service in Services
    if (Match(service.Profile،S.Profile))
      return true
   return false
```

21. البحث عن خدمة مركّبة:

الدخل: الخدمة المُراد البحث عنها.

الخرج: نتيجة البحث المنطقية.

الخوارزمية:

```
//CS typeof CompositeService

findCompositeService(CS)
  foreach cSservice in Services
    if cService is composite
        if (Match(cService.Profile،CS.Profile))
              return rrue
   return false
```

22. حذف خدمة:
الدخل: الخدمة المُراد حذفها.
الخرج: النتيجة المنطقية لعميلة الحذف.
الخوارزمية:

```
//S typeof Service

deleteService(S)
    if (findService(S))
        Services ← Services \ {S}
        return true
    else
        return false
```

23. حذف خدمة مركّبة:
الدخل: الخدمة المركّبة.
الخرج: النتيجة المنطقية للحذف.
الخوارزمية:


```
//CS typeof CompositeService

deleteCompositeService(CS)
    if (findService(CS))
        Services ← Services \ {CS}
        foreach service in CS.Services
            ServiceCatalogue(S)←
                ServiceCatalogue(S) \ {CS}
        return true
    else
        return false
```

24. تنفيذ خدمة:
الدخل: الخدمة.
الخرج: خرج الخدمة المُنفَّذة.
الخوارزمية:

```
//S typeof Service

executeService(S)
    if (findService(S))
        InvokeWebService(S.G.Address،S.Parameters)
```

```
        return S.output
    else
     return Null
```

25.تنفيذ خدمة مركّبة باستخدام مخدّم مكامل الخدمات "Service Integrator Server":

الدخل: الخدمة المركّبة.

الخرج: خرج الخدمة المركّبة.

الخوارزمية:

```
//CS typeof CompositeService

executeCompositeService(CS)
  if (findService(CS))
    foreach service in CS.Services
     MatchedServices ← {(service،

Match(service.RequestDelegation))}
    foreach service in MatchedServices
            executeService(service)
            updateServiceParameters(CS)
    return CS.output
  else
    return Null
```


26. مجموعة توابع المطابقة:

```
//request typeof Profile

Match(request)
     recordMatch empty list;
     foreach adv in advertisements
          if match(request،adv)
            Recordmatch.add(adv)
return recordmatch

//request typeof Profile
//advertisment typeof Profile

Match(request،advertisment)
     inputMatched =InputMatch(request،
                                advertisment)
     outputMatched =outputMatch(request،
                                advertisment)
    if (inputMatched<>Null)and
       (outputMatched<>Null)
```

```
        return true
    else
        return false

//outputsRequest listofType Parameter
//OutputAdvertisment listofType Parameter

ouputMatch(outputsRequest،OutputAdvertisment)
    Globaldegreematch=Exact
    Degreematch=Null
    foreach outR in outputsRequest
        foreach outA in outputsAdvertisment
            Degreematch=maxdegreematch(outR،outA)
            if (Degreematch<>Null)
                exit for
        if  (Degreematch=Null)
            return Null
        if (Degreematch<globaldegreematch)
            Globaldegreematch=degreematch
    return Globaldegreematch

//inputsRequest listofType Parameter
//inputAdvertisment listofType Parameter

inputMatch(inputsRequest،inputAdvertisment)
    Globaldegreematch=Exact;
    foreach inR in inputsRequest
        foreach inA in inputsAdvertisment
            Degreematch=maxdegreematch(inR،inA)
            if (Degreematch<>Null)
                exit for
        if (Degreematch=Null)
            return Null
        if (Degreematch<globaldegreematch)
            Globaldegreematch=degreematch
    return Globaldegreematch

//inR typeof Parameter
//inA typeof Parameter

degreeOfMatch(inR،inA)
    if inR=inA
        return Exact
    if inR subClassOf inA
        return Exact
    else
      return Null
```

6-5. خلاصة

قمنا من خلال هذا الفصل بتقديم النموذج العملياتي المقابل للنموذج النظري لنظم المعلومات خدمية التوجه الدلالية S_3OS الذي استعرضناه في الفصل الخامس، وذلك من خلال تقديم مجموعة التعاريف والخوارزميات اللازمة لعمل المكوّنات الأساسية في هذا النموذج إنطلاقاً من نشر الخدمات واستكشافها إلى مكاملتها فيما بينها بهدف تقديم خدمات جديدة.

تطبيق عملي

Use Case

مقدّمة:

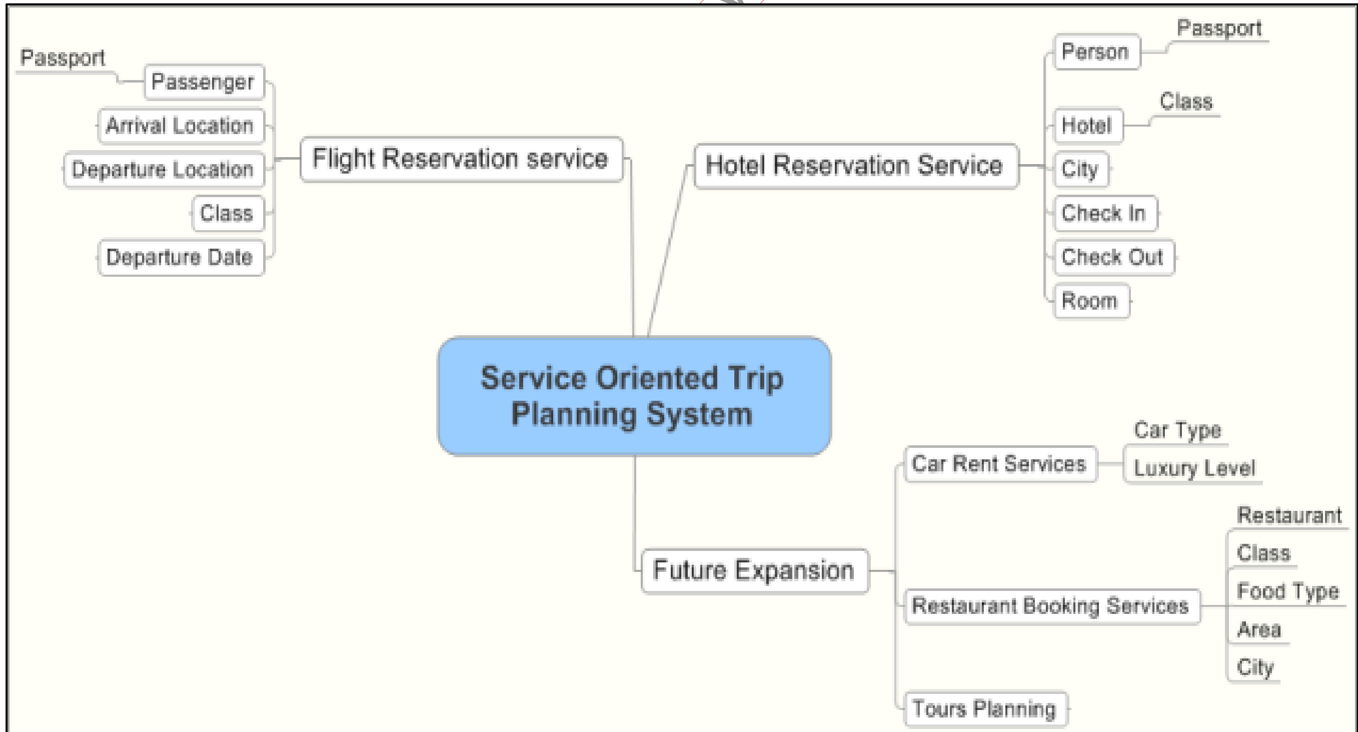
نقوم من خلال هذا الفصل باستعراض حالة عملية لبناء نظام معتمد على الخدمات من خلال تطبيق النموذج الذي قدمناه من خلال الفصول السابقة S_3OS، سنأخذ نظام التخطيط للرحلات (Trip Planning) الذي تم استخدامه في مجمل الأمثلة التي شرحنا من خلالها مختلف المفاهيم في الفصول السابقة. وسنقتصر في هذا المثال على تقديم خدمتين جزئيتين أساسيتين وهما خدمة حجز رحلات الطيران إلى جانب الحجز الفندقي، هذا ومن الممكن أن يشتمل هذا النظام على مجموعة أخرى من الخدمات كخدمة استئجار سيارة ما خلال الرحلة، أو التخطيط لمجموعة من الرحلات السياحية خلال هذه الرحلة أو حتى خدمة حجوزات المطاعم وغيرها من الخدمات الأخرى. هذا وسنقوم بطرح آلية بناء هذا النظام بالطريقة التقليدية لبناء النظم خدمية التوجّه وذلك لإتاحة الفرصة للمقارنة بين النموذجين من الناحية العملية.

1-7. نظام التخطيط للرحلات الالكتروني:

وهو عبارة عن موقع الكتروني يقوم وبالنيابة عن المستخدم بالقيام بالحجز لرحلة الطيران المناسبة والقيام بالحجز الفندقي الموافق من خلال البحث عن مزوّدي هذه الخدمات بما يتناسب مع المعايير والمواصفات التي تتلائم مع رغبات المستخدم من حيث التاريخ والوقت المناسب وجودة الخدمة وتصنيفها.

يحتوي هذا الموقع على صفحة تسمح للمستخدم بتحديد متطلباته مثل تاريخ السفر ومكان الإنطلاق ومكان الوصول ودرجة الحجز باستخدام الطائرة والوقت المناسب للحجز الفندقي وتصنيف الفندق وتاريخ نهاية الحجز وغيرها. وإذا رغبنا بتوسيع الخدمات التي يقدمها هذا النظام فيمكن للمستخدم تحديد أنواع الطعام التي يحبها ليقوم النظام بالقيام بالحجوزات المناسبة في المطاعم الصينية أو المطاعم التي تشتهر بتقديم الطعام النباتي مثلاً في مكان الإقامة. طبعاً ومن متطلبات هذا النظام أن يكون متاحاً في أي وقت. يوضح الشكل "1-7" مخطط من جلسة العصف الذهني " Brain Storming" لجمع متطلبات هذا النظام.

بالطبع يمكن بناء هذا النظام من الصفر، ويمكن بناؤه بالاعتماد على خدمات الوب التي ينشرها مزوّدي الخدمات التي يتألف منها هذا النظام على الانترنت أي أن يكون النظام خدمي التوجّه، سنستعرض فيما يلي كيفية بناء هذا النظام بطريقة معتمدة على الخدمات ومن ثم نقوم بإدخال البعد الدلالي باستخدام نموذج الـS_3OS.



الشكل 1-7، مخطط من جلسة العصف الذهني لجمع متطلبات نظام التخطيط للرحلات

2-7. الطريقة التقليدية لبناء النظام بالاعتماد على الخدمات:

نذكر أهم النقاط والخطوات المتعلقة ببناء هذا النظام بالإعتماد على مجموعة موجودة من الخدمات على الانترنت:

1- لعل أهم الخطوات التي تواجه مهندس البرمجيات هي في البحث عن حزمة من خدمات الوب المنتشرة على الانترنت بحيث تغطي احتمالات أكبر من المواصفات التي تقدمها هذه الخدمات بالإضافة إلى الأماكن أو المدن التي تقدم هذه الأطراف خدماتها فيها، فعلى سبيل المثال على مهندس البرمجيات البحث عن خدمات حجز فندقي بتصنيف ثلاث نجوم، وخدمات حجز فندقي أخرى بتصنيف أربع وخمس نجوم. نُذكِّر بأن البحث عن خدمات الوب تتم من خلال دليل خدمات الوب على الانترنت "UDDI" الذي تحدثنا عنه في الفصل الثاني.

2- يجب مراعاة إيجاد مثل هذه الخدمات التي تتيح إمكانية الحجز الفندقي في أكبر عدد من البلدان والمدن.

3- بشكل مشابه يتم البحث عن أكبر عدد ممكن من خدمات الوب التي تقدم خدمة حجز رحلات الطيران بحيث تغطي أكبر عدد ممكن من البلدان أيضاً. فعلى سبيل المثال يعرض كل من الشكلين "2-7"، "3-7" مجموعة من الخدمات التي وجدناها على الانترنت مع عناوينها.

خدمات الوب لحجز رحلات الطيران

مزود الخدمة	عنوان الخدمة
keyfortravel.com	http://ws.keyfortravel.com/webservices/K4TAirSell.asmx?wsdl
webxml.com.cn	http://www.webxml.com.cn/web...es/DomesticAirline.asmx?wsdl
95022.com	http://www.95022.com/AirticketWebService.asmx?WSDL

الشكل 2-7،جدول يحتوي قائمة بخدمات وب تتيح الحجز لرحلة طيران.

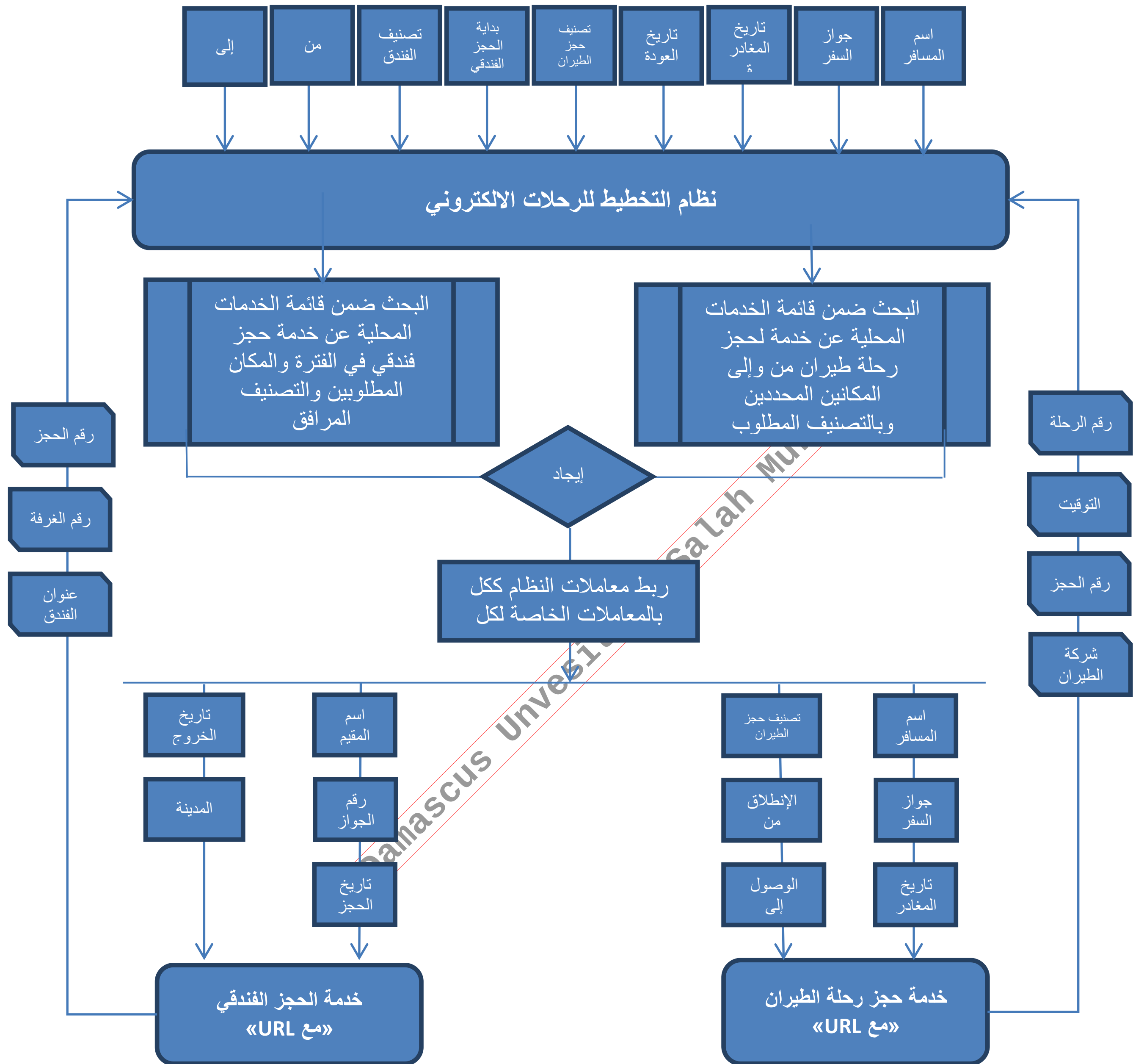
خدمات الوب للحجز الفندقي

مزود الخدمة	عنوان الخدمة	البلد
hotels-europe.com	http://www.hotels-europe.com/webservices/General.asmx?WSDL	أوروبا
keyfortravel.com	http://ws.keyfortravel.com/hotel/hotelsell.asmx?wsdl	البرتغال
touricoholidays.com	http://ws1.touricoholidays.com/HotelsService.asmx?wsdl	الولايات المتحدة
allysoft.ru	http://www.allysoft.ru/XML/Connector.asmx?WSDL	روسيا
hotellinx.com	http://www.hotellinx.com/Web...HotellinxXmlService.asmx?WSDL	فنلندا

الشكل 3-7، جدول يحتوي قائمة بخدمات وب تتيح إمكانية الحجز الفندقي لمزوّدي خدمات من بلدان عدّة.

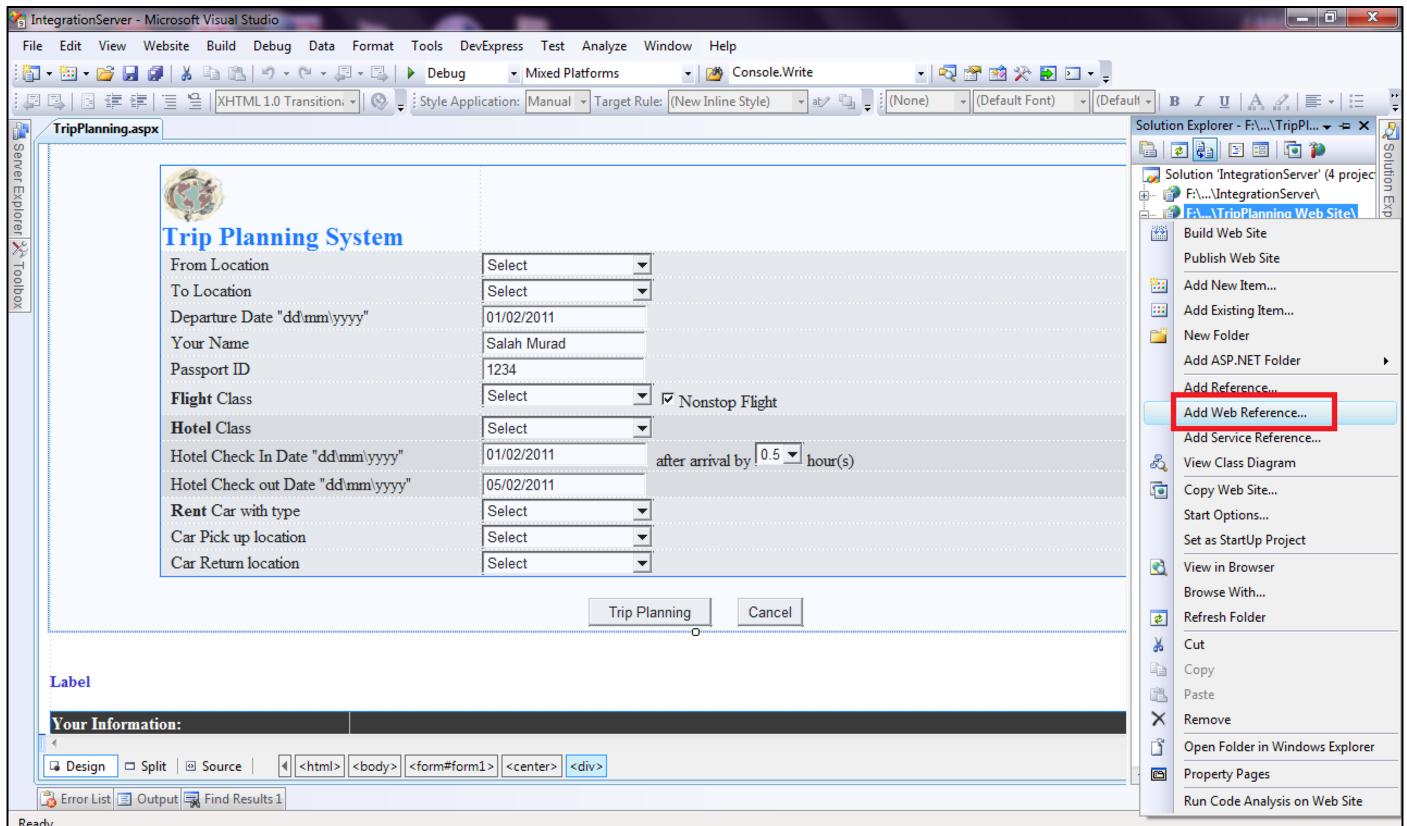
4- يتم إعداد التصميم الداخلي لبنية النظام وآلية اختيار الخدمات المناسبة لرغبات المستخدمين وتحديد آلية تفاعل هذه الخدمات بعضها مع بعض وبشكل برمجي بهدف إنجاز المهمة الكلية للنظام.

5- إعداد طريقة تخاطب بارامترات الخدمات بجميع الاحتمالات الموجودة مع مراعاة اختلاف تسمية هذه البارامترات ونوعها ونمط معطياتها من خدمة إلى أخرى. كما هو موضح بالشكل "7-4".

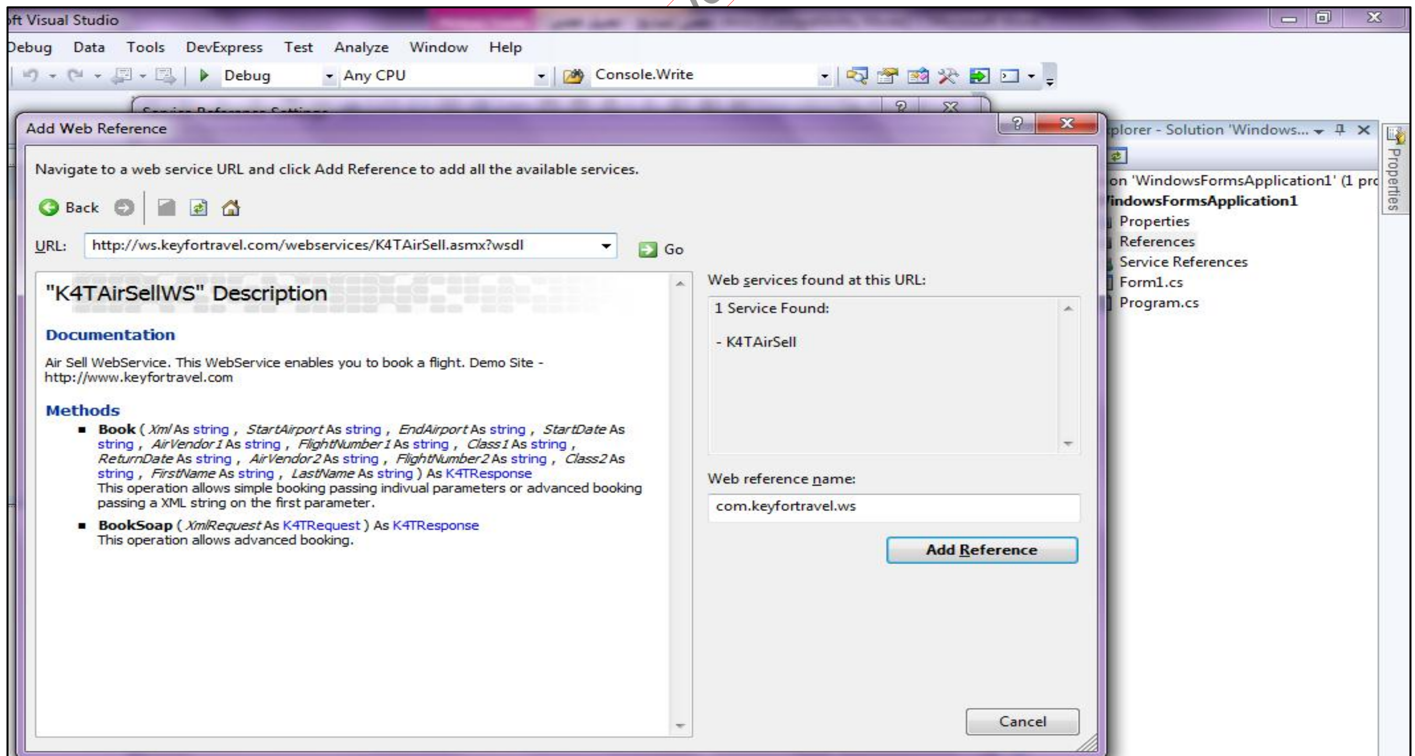


الشكل 7-4، البنية الداخلية لنظام التخطيط للرحلات

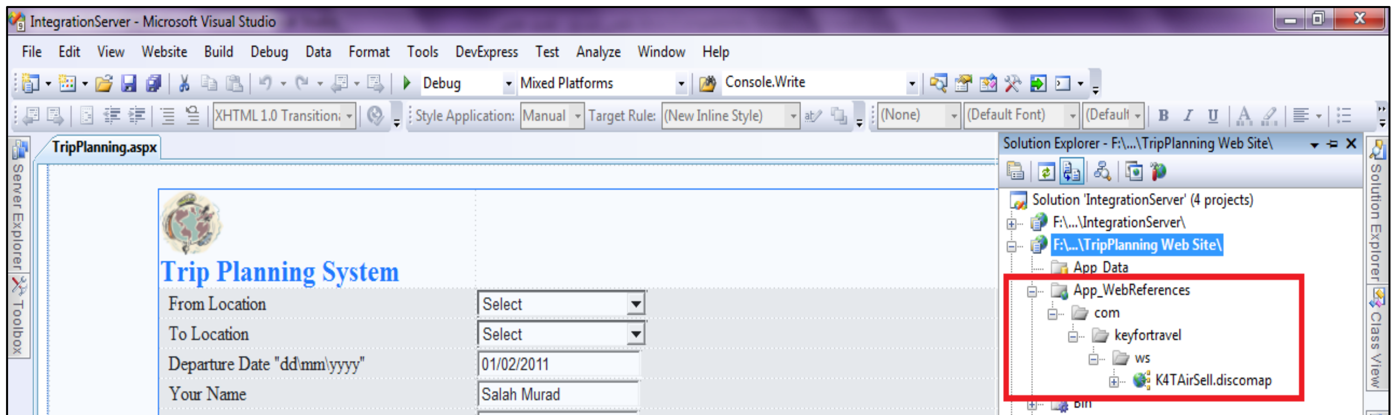
6- يجري استخدام الخدمات التي تم العثور عليها بالاعتماد على العنوان الذي تتواجد فيه على الانترنت وبشكل ثابت ضمن الرماز البرمجي أو ضمن قاعدة بيانات خاصة بالنظام. فعلى سبيل المثال توضح الأشكال "7-5-أ" و"7-5-ب" و"7-5-ج" عملية إضافة خدمة وب من خلال عنوانها على الانترنت ضمن بيئة تطوير التطبيقات Microsoft Visual Studio.



الشكل 7-5-أ، طلب إضافة مرجع لخدمة وب



الشكل 7-5-ب، إضافة مرجع لخدمة وب باستخدام عنوانها



الشكل 7-5-ج، تمت إضافة مرجع خدمة الوب

7- يتم تصميم وبناء واجهات النظام المُوجَّهة لكل من المستخدم النهائي ولمدير النظام.

8- يتم نشر هذا النظام على الانترنت ليكون متاحاً للمستخدمين بشكلٍ مباشر.

9- من المهم هنا أن نشير إلى أنه في حال إنشاء خدمات فندقية بتصنيف ست نجوم على سبيل المثال فعلى مهندس البرمجيات القيام بإجراء التعديل البرمجي اللازم على النظام من جديد بعد البحث عن مثل هذه الخدمات على الوب وإضافتها إلى النظام وإلا فسيكون النظام قاصر عن مواكبة آخر التحديثات.

10- يجب إعادة تحديث النظام بشكل يدوي من خلال تحديث قائمة الخدمات لاسيما عند إلغاء بعض خدمات الوب التي يعتمد عليها النظام حالياً.

11- يجب إعادة تحديث النظام وبتدخل يدوي عند حدوث أي تغيير على طريقة استخدام الخدمة التي يعتمد عليها النظام.

12- أي تغيير في عناوين الخدمات المستخدمة في النظام يتطلب تحديث هذه العناوين.

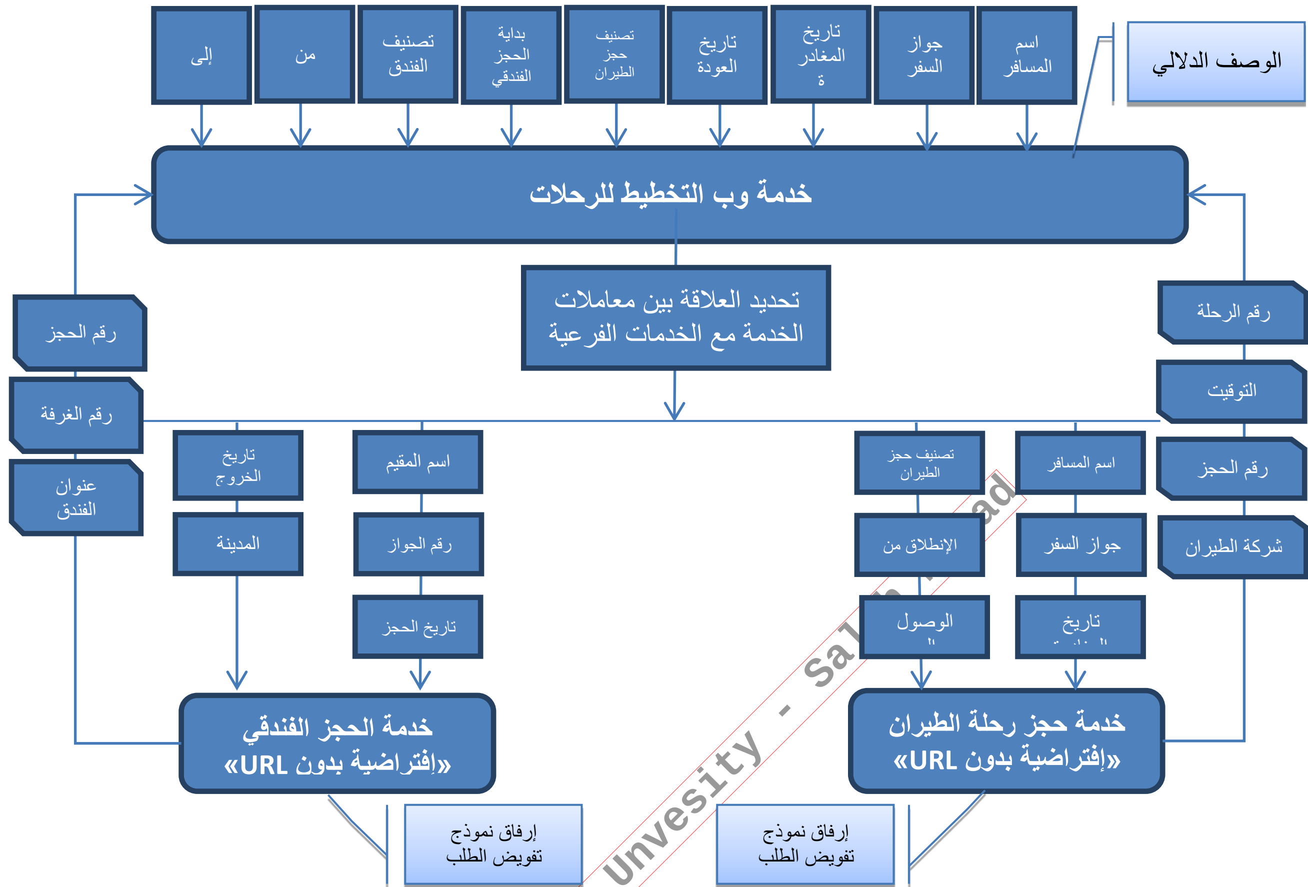
7-3. بناء النظام باستخدام الخدمات الموصّفة دلالياً من خلال النموذج S_3OS:

نذكر أهم النقاط والخطوات المتعلقة ببناء هذا النظام بالإعتماد على مجموعة موجودة من الخدمات الموصّفة دلالياً:

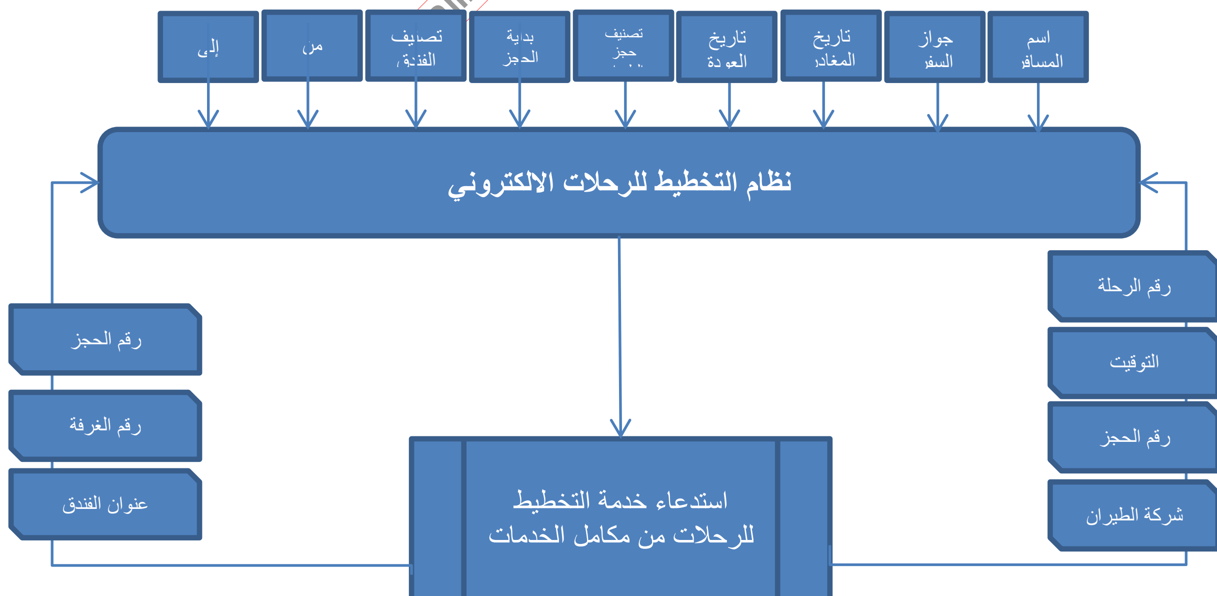
1- يعتمد بناء النظام بشكل أساسي على إيجاد الخدمات الجزئية من خدمة الحجز الفندقي وحجز رحلات الطيران والتي تم إرفاقها بتوصيف دلالي، بحيث يتم التأشير إليها من خلال وصفها بدلاً من استخدام عنوانها المحدّد والثابت كما في الطريقة التقليدية.

2- تصميم إجرائية العمل لخدمة افتراضية للتخطيط للرحلات تتألف من خدمتين للحجز الفندقي وحجز الطيران وتزويد كلٍ منهما بنموذج تفويض الطلب والذي يمثّل التوصيف الدلالي لإمكانيات هذه الخدمات كما هو موضّح بالشكل "7-6-أ" والشكل"7-6-ب". إن نموذج تفويض الطلب هذا سيحوّل الخدمة ذاتها إلى طالب لخدمة مكافئة لها أثناء التنفيذ باستخدام أي مكامل الخدمات يتم تطويره بما يحقق توصيف النموذج S_3OS، وهو مخدم يتم ترفيعه على الانترنت ويكون متاحاً لمزوّدي الخدمات لإضافة خدمات الوب التي يقدمونها إلى جانب توصيفها الدلالي، كما ويسمح بتصميم إجراءات عمل أو سيناريوهات مكاملة مختلفة بين الخدمات الموجودة في مستودع الخدمات أو بين خدمات افتراضية، يمكن الرجوع إلى الفصلين الخامس والسادس للإطلاع على تفاصيل ودور مكامل الخدمات من خلال كلٍ من النموذج النظري والرياضي والعملياتي. لقد قمنا

بتطوير مكامل الخدمات الخاص بنا في إطار مثال نظام التخطيط للرحلات الالكتروني الذي نعرضه في هذا الفصل.



الشكل 7-6-أ، بنية خدمة التخطيط للرحلات الدلالية



الشكل 7-6-أ، بنية نظام التخطيط للرحلات الالكتروني

3- يتم تصميم وبناء واجهات النظام المُوجَّهة لكل من المستخدم النهائي ولمدير النظام.

4- يتم نشر هذا النظام على الانترنت ليكون متاحاً للمستخدمين بشكلٍ مباشر.
5- ليس على مهندس البرمجيات متابعة حالة الخدمات الجزئية التي استخدمها في بنائه لنظام التخطيط للرحلات، حيث سيقوم مكامل الخدمات بالبحث عن الخدمة الفعلية المطابقة لهذه الخدمات الجزئية بناءً على وصفها الدلالي، كما سيقوم بمحاولة البحث عن خدمة رديفة في حال وجود أية مشكلة في تنفيذ الخدمة التي تم العثور عليها سابقاً وهكذا.
6- ليس على مهندس البرمجيات إجراء أي تحديث على نظامه فيما لو تغيّرت عناوين خدمات الوب الخاصة بالحجز الفندقي مثلاً أو إذا تغيّرت طريقة استخدامها، لأنه لم يعتمد على عناوينها في بناء نظامه وإنما على وصفها الدلالي.

4-7. وظائف مكامل الخدمات الدلالية:

سنعرض فيما يلي أهم الوظائف التي يتيحها مكامل الخدمات الدلالية من حيث نشر الخدمات وتوصيفها وبناء إجراءات العمل أو سيناريوهات المكاملة الخاصة بهذه الخدمات.

1- إضافة خدمة وب للتخطيط للرحلات:

الشكل 7-7، إضافة خدمة باستخدام مكامل الخدمات

2- تحديد معاملات الخدمة المُضافة:

TripPlanning - S3OS - Windows Internet Explorer

http://localhost:2151/defaultVertical.aspx?ShortcutViewID=Process_DetailView&ShortcutObjectKey=6&ShortcutObjectClassName=ServiceProvider.Moc

Service Registeration
- Service Provider
- Service
- Process
- Process Parameter
- Process Description
- Process Request Template
- Semantic Description

Ontology Management

Reports

Users

Trip Planning / Process Parameter / FromLocation / Process / TripPlanning

Process

New | Delete | Edit | Previous Object | Next Object | Refresh

Process

Web Method: Method0
Parent ID: DefaultProcess
Process Order: 0
Row ID: 6
Process Name: TripPlanning
Service ID: Trip Planning
Request Template ID: request template for the composite service trip planning
Virtual Process: True
Source Service ID: DefaultService

Description Of Process | Parameters Of Process

New | Link | Unlink | Delete | Edit

Process ID	Parameter Name	Parameter Order	Parameter Type	Parameter Identifier	Source Parameter ID
TripPlanning	FromLocation	1	System.String	In	
TripPlanning	ToLocation	2	System.String	In	
TripPlanning	DepartureDate	3	System.DateTime	Out	
TripPlanning	HotelClass	4	System.String	In	
TripPlanning	HotelFromDate	5	System.DateTime	In	
TripPlanning	HotelToDate	6	System.DateTime	In	
TripPlanning	ClientName	7	System.String	In	
TripPlanning	RoomID	8	System.Int32	Out	

Local intranet | Protected Mode: Off 100%

الشكل 7-8، إضافة خدمة باستخدام مكامل الخدمات

3- توصيف نموذج تفويض الطلب لخدمة الحجز الفندقي التي تنتمي إلى الخدمة الكلية للتخطيط للرحلات:

Quick create...

Service Registeration
- Service Provider
- Service
- Process
- Process Parameter
- Process Description
- Process Request Template
- Semantic Description

Ontology Management

Reports

Users

request template of Five Star ... / Process / HotelReservation / request template for virtual h... / request template for virtual h...

Semantic Description Header

New | Delete | Edit | Previous Object | Next Object | Refresh

Row ID: 11
Semantic Description Header Name: request template for virtual hotel reservation in trip planning

Semantic Description Details

New | Link | Unlink | Delete | Edit

Subject	Subject Type	Predicate	_Object	Object Type
HotelClass	Class	hasValue	FourStars	Value
Location	Class	hasValue	AnyLocation	Value

Items per page: 20

Service Provider | Service | Process | Process Parameter | Process Description | Process Request Template | Semantic Description | Concepts | Relations |

Done Local intranet | Protected Mode: Off 100%

الشكل 7-9، توصيف نموذج تفويض الطلب

4- تحديد معاملات أحد الخدمات الجزئية وعلاقتها بمعاملات الخدمة الأساسية:

HotelReservation - S3OS - Windows Internet Explorer

http://localhost:2151/defaultVertical.aspx?ShortcutViewID=Process_DetailView&ShortcutObjectKey=4&ShortcutObjectClassName=ServiceProvider.Moc

Service Registeration
- Service Provider
- Service
- Process
- Process Parameter
- Process Description
- Process Request Template
- Semantic Description

Ontology Management

Reports

Users

request template for virtual h... / Process Parameter / VisitorName / Process / HotelReservation

Process

New | Delete | Edit | Previous Object | Next Object | Refresh

Process

Web Method: Method2
Parent ID: TripPlanning
Process Order: 2
Row ID: 4
Process Name: HotelReservation

Service ID: Trip Planning
Request Template ID: request template for virtual hotel reservation in trip planning
Virtual Process: True
Source Service ID: DefaultService

Description Of Process | Parameters Of Process

New | Link | Unlink | Delete | Edit

Process ID	Parameter Name	Parameter Order	Parameter Type	Parameter Identifier	Source Parameter ID
HotelReservation	VisitorName	1	System.String	In	ClientName
HotelReservation	HotelFromDate	2	System.DateTime	In	HotelFromDate
HotelReservation	HotelToDate	3	System.DateTime	In	HotelToDate
HotelReservation	RoomID	4	System.Int32	Out	RoomID
HotelReservation	HotelReservationID	5	System.Int32	Out	HotelReservationID
HotelReservation	HotelName	6	System.String	Out	HotelName
HotelReservation	HotelPhone	7	System.String	Out	HotelPhone
HotelReservation	HotelURI	8	System.String	Out	HotelURI

Done | Local intranet | Protected Mode: Off | 100%

الشكل 7-10، معاملات الخدمة الجزئية

5- توصيف خدمة الحجز الفندقي التي تنتمي إلى الخدمة الكلية للتخطيط للرحلات:

Quick create...

Service Registeration
- Service Provider
- Service
- Process
- Process Parameter
- Process Description
- Process Request Template
- Semantic Description

Ontology Management

Reports

Users

Service / Mosco Hotel Reservation / MoscoHotelReservation / 5 / description for Mosco Hotel Pr...

Semantic Description Header

New | Delete | Edit | Previous Object | Next Object | Refresh

Row ID: 9
Semantic Description Header Name: description for Mosco Hotel Process

Semantic Description Details

New | Link | Unlink | Delete | Edit

Subject	Subject Type	Predicate	_Object	Object Type
HotelClass	Class	hasValue	FourStars	Value
Location	Class	hasValue	Moscow	Value

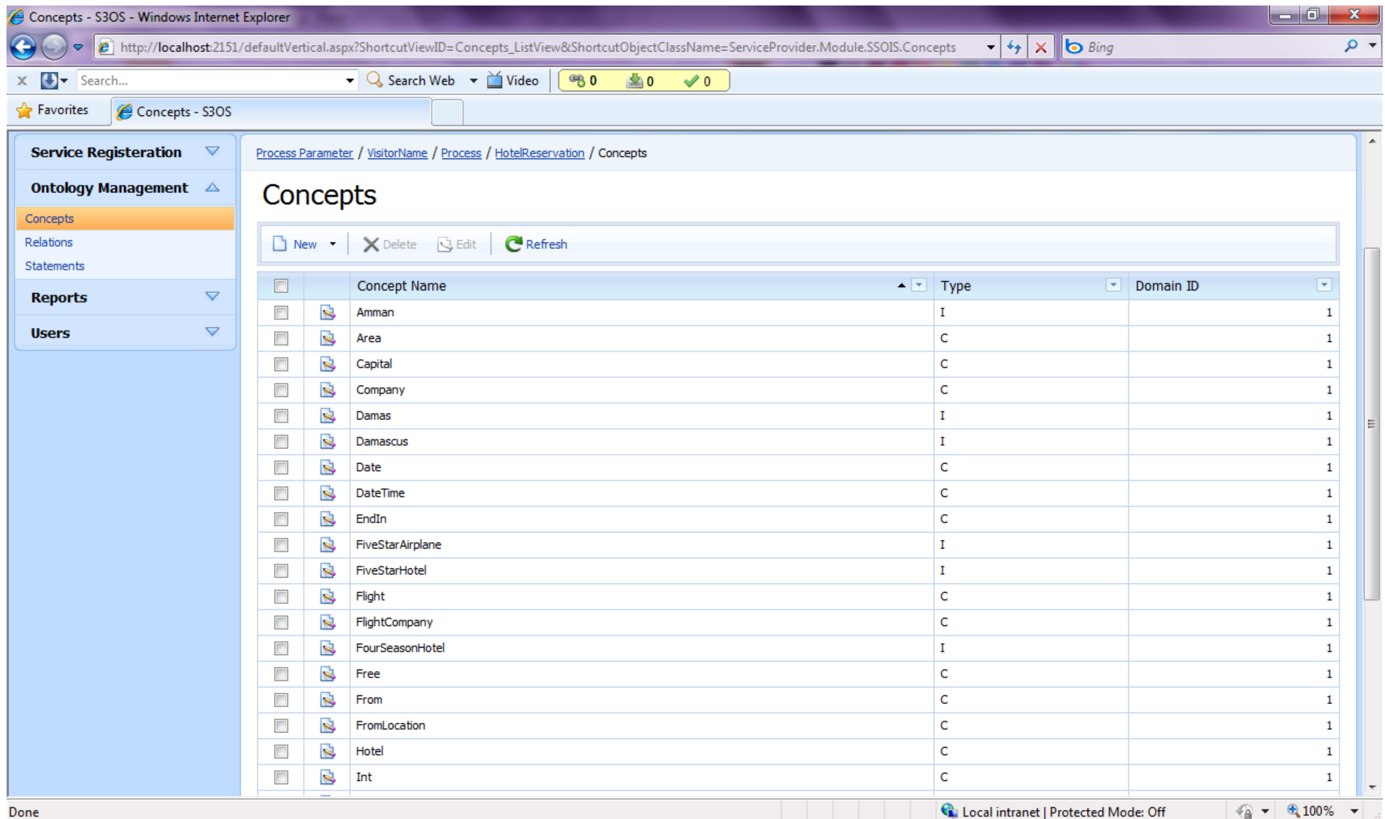
Items per page: 20

Service Provider | Service | Process | Process Parameter | Process Description | Process Request Template | Semantic Description | Concepts | Relations |

Done | Local intranet | Protected Mode: Off | 100%

الشكل 7-11، توصيف خدمة الحجز الفندقي في موسكو

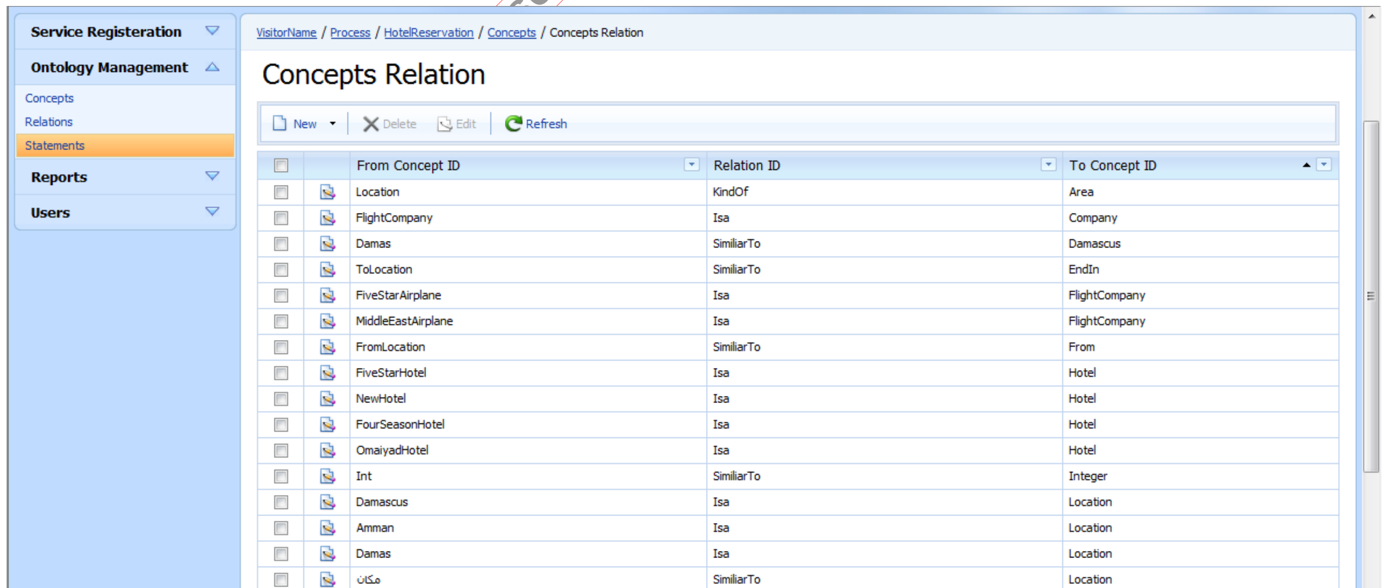
6- إضافة المفاهيم إلى الأنطولوجية:



Concept Name	Type	Domain ID
Amman	I	1
Area	C	1
Capital	C	1
Company	C	1
Damas	I	1
Damascus	I	1
Date	C	1
DateTime	C	1
EndIn	C	1
FiveStarAirplane	I	1
FiveStarHotel	I	1
Flight	C	1
FlightCompany	C	1
FourSeasonHotel	I	1
Free	C	1
From	C	1
FromLocation	C	1
Hotel	C	1
Int	C	1

الشكل 7-12، مفاهيم الأنطولوجية

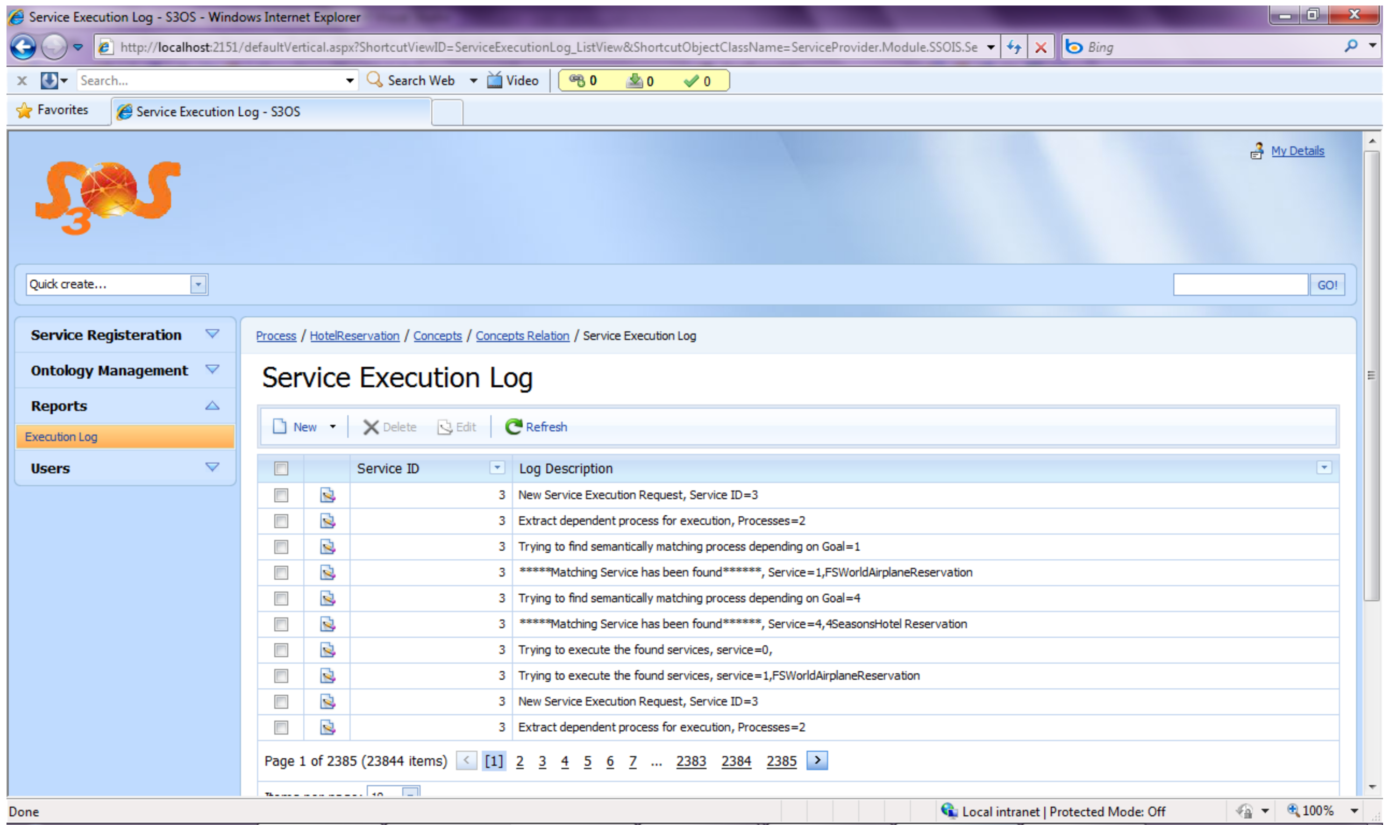
7- العلاقات بين المفاهيم ضمن الأنطولوجية:



From Concept ID	Relation ID	To Concept ID
Location	KindOf	Area
FlightCompany	Isa	Company
Damas	SimiliarTo	Damascus
ToLocation	SimiliarTo	EndIn
FiveStarAirplane	Isa	FlightCompany
MiddleEastAirplane	Isa	FlightCompany
FromLocation	SimiliarTo	From
FiveStarHotel	Isa	Hotel
NewHotel	Isa	Hotel
FourSeasonHotel	Isa	Hotel
OmaiyadHotel	Isa	Hotel
Int	SimiliarTo	Integer
Damascus	Isa	Location
Amman	Isa	Location
Damas	Isa	Location
مكان	SimiliarTo	Location

الشكل 7-13، العلاقات في الأنطولوجية

8- سجل تنفيذ الخدمات: ويظهر الخطوات التي يتبعها مكامل الخدمات أثناء تنفيذ أحد الخدمات من مستودع الخدمات.

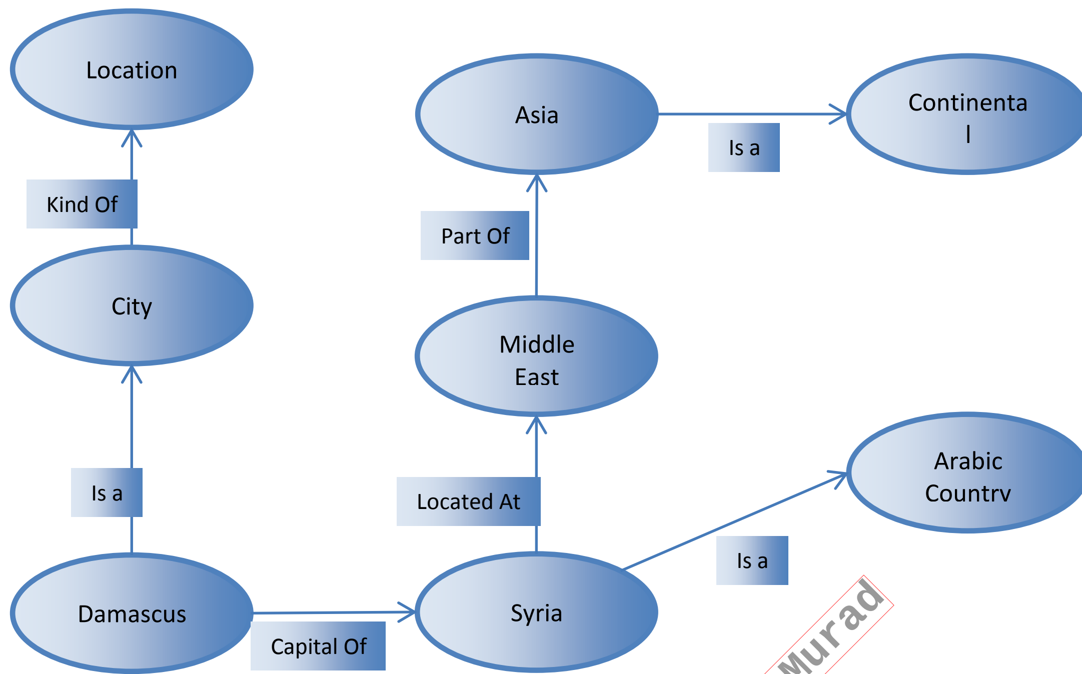


الشكل 7-14، سجل التنفيذ في مكامل الخدمات

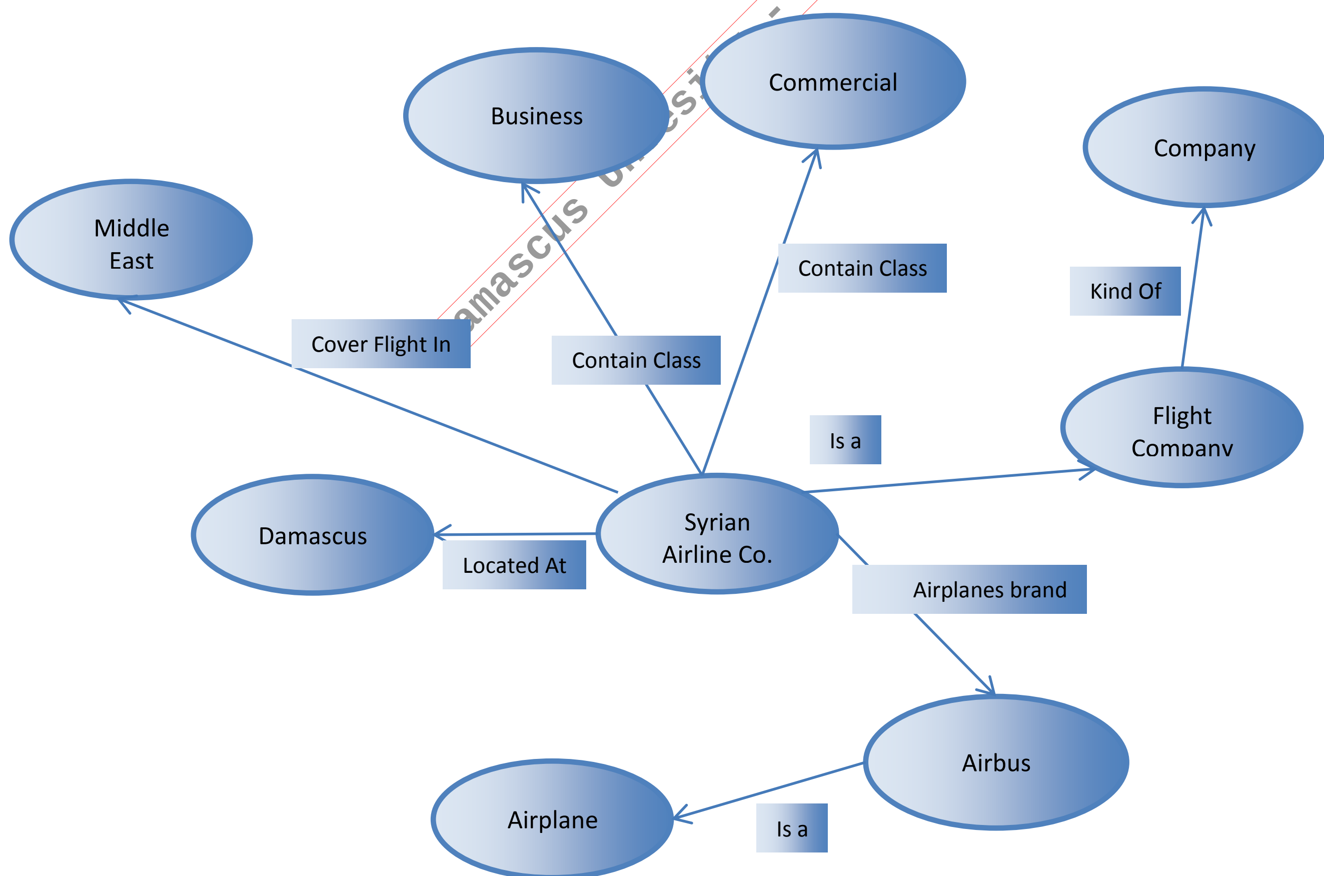
7-5. حالات اختبار:

يتطلب مكامل الخدمات في مثالنا عن نظام التخطيط للرحلات الالكتروني بالخدمات التي يحتويها إلى ثلاثة أنواع من الأنطولوجيات و هي الأنطولوجية الجغرافية، أنطولوجية الفنادق، أنطولوجية شركات ورحلات الطيران ويُطلَق على هذه الأنطولوجيات بأنطولوجيات مجال أو ميدان العمل " Domain Ontologies". حيث تختص الأنطولوجية الجغرافية بالمدن والبلدان والأماكن والعواصم والمسافات بين المدن ومعلومات عن الجبال والأنهار وغيرها من المعلومات المتعلقة بالأماكن الجغرافية، أما أنطولوجية الفنادق فتحتوي على معلومات تتعلق بسلاسل الفنادق وأماكن تواجدها وعناوينها وسعاتها وتصنيفها، أما أنطولوجية شركات الطيران فتحتوي على معلومات تتعلق بشركات الطيران ونوعها وتصنيفاتها والأماكن التي تغطيها وغيرها من المعلومات. يقوم مكامل الخدمات بالإستعلام ضمن هذه الأنطولوجيات عن الموارد التي تحقق مواصفات معينة.

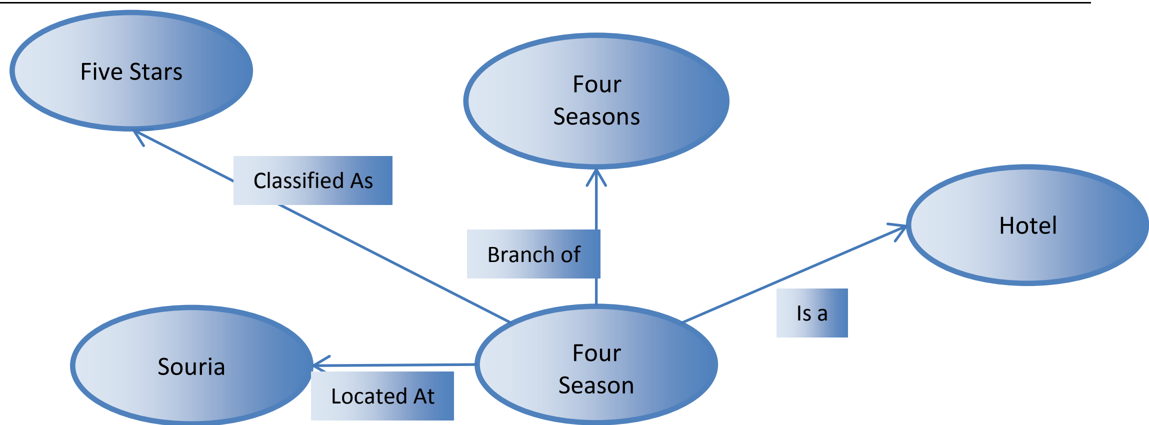
قبل البدء بتطبيق بعض الأمثلة حول استخدام النظام، نعرض فيما يلي مثالاً عن المعلومات التي تحتويها هذه الأنطولوجيات والتي تشبه ما تم استخدامه في أمثلتنا. حيث يعرض المخطط "7-15" جزء من الأنطولوجية الجغرافية المتعلقة بسوريا، والمخطط "7-16" يحتوي جزء من أنطولوجية الفنادق المتعلقة بفندق الفصول الأربعة في سوريا، والمخطط "7-17" يحتوي جزء من أنطولوجية رحلات الطيران المتعلقة بالشركة السورية للطيران.



الشكل 7-15، جزء من الأنطولوجية الجغرافية



الشكل 7-16، جزء من أنطولوجية رحلات وشركات الطيران



الشكل 7-17، جزء من أنطولوجية الفنادق

نلاحظ وجود اختلاف في التعبير عن نفس المفهوم "Syria" و"Souria" وبطريقتين مختلفتين في أنطولوجية الفنادق وأنطولوجية الأماكن الجغرافية، لذلك يجب فضّ هذا الاختلاف بحيث يتم فهمهما على أنهما يعبّران عن نفس الشيء، وهو ما يسمى بالوساطة بين الأنطولوجيات " Ontology Mediation" كما في الشكل "7-18".



الشكل 7-18، الوساطة بين الأنطولوجيات

سنقوم من خلال الجدول التالي باستعراض الخدمات الموجودة في مستودع الخدمات إلى جانب المواصفات المتعلقة بكلٍ منها:

الخدمة	مزوّد الخدمة	الوصف	عنوان الخدمة
FourSeasons Hotel Reservation	Four Seasons Hotel and Resorts	- خدمة للحجز الفندقي - فندق خمس نجوم - إمكانية الحجز في أية مدينة حول العالم - إمكانية الحجز بين تاريخين	http://….
Moskova Hotel Reservation	Moscow Hotels	- خدمة للحجز الفندقي - فندق أربع نجوم - إمكانية الحجز في مدينة موسكو فقط - إمكانية الحجز بين تاريخين	http://….
Sheraton Hotel Booking	Sheraton Hotels	- خدمة للحجز الفندقي - فندق خمس نجوم - إمكانية الحجز في أية مدينة حول العالم - إمكانية الحجز بين تاريخين	http://….
Airline Reservation	Malaysia Airways	- خدمة حجز رحلات الطيران - إمكانية الحجز من أي مكان في العالم وإلى أي مكان - إمكانية الحجز بالدرجة الإقتصادية، درجة رجال الأعمال، والدرجة الأولى	http://….
Flight Reservation	Middle East Airlines	- خدمة حجز رحلات الطيران - إمكانية الحجز من أي مكان في العالم وإلى أي مكان - إمكانية الحجز بالدرجة الإقتصادية، درجة رجال الأعمال، والدرجة الأولى	http://….

قمنا بتنفيذ مجموعة من حالات الاستخدام لنظام التخطيط للرحلات الالكتروني وباستخدام مجموعة مختلفة من المُعاملات مع مراعاة حالات متعدّدة للخدمات وحصلنا على نتائج الاستخدام الصادرة عن مكامل الخدمات بما يتوافق مع رغبات المستخدم وحالات الخدمات وذلك بالشكل الآتي:

- شاشة الاستخدام الرئيسية:

Untitled Page - Windows Internet Explorer
http://localhost:1750/TripPlanning%20Web%20Site/TripPlanning.aspx

Trip Planning System

From Location: Select
To Location: Select
Departure Date "dd\mm\yyyy"
Your Name
Passport ID
Flight Class: Select
Hotel Class: Select
Hotel Check In Date "dd\mm\yyyy" after arrival by 0.5 hour(s)
Hotel Check out Date "dd\mm\yyyy"

Trip Planning Cancel

الشكل 7-19، الصفحة الرئيسية لنظام التخطيط للرحلات الالكتروني

- الحالة الأولى:

اسم المعامل	قيمة المعامل	حالة الخدمات		النتيجة
اسم المسافر	صلاح مراد	Up	FourSeasons Hotel Reservation	☑
من مدينة	دمشق	Up	Moskova Hotel Reservation	
إلى مدينة	بيروت	Up	Sheraton Hotel Booking	
تاريخ السفر	01/01/2012	Up	Airline Reservation	☑
لغاية	06/01/2012	Up	Flight Reservation	
درجة الحجز بالطائرة	الدرجة الأولى	الشرح: يقوم مكامل الخدمات بالبحث ضمن مستودع الخدمات بشكل عشوائي عن خدمات تحقق المواصفات المطلوبة.		
تصنيف الفندق	خمس نجوم			
بداية حجز الفندق	01/01/2012			
نهاية حجز الفندق	06/01/2012			

- الحالة الثانية:

اسم المعامل	قيمة المعامل	حالة الخدمات		النتيجة
اسم المسافر	رامي	Up	FourSeasons Hotel Reservation	☑
من مدينة	دمشق	Up	Moskova Hotel Reservation	
إلى مدينة	الرياض	Up	Sheraton Hotel Booking	
تاريخ السفر	01/01/2012	Down	Airline Reservation	
لغاية	06/01/2012	Up	Flight Reservation	☑
درجة الحجز بالطائرة	الدرجة الأولى	الشرح: يقوم مكامل الخدمات بالبحث عن خدمة حجوزات طيران بديلة للخدمة المعطّلة.		
تصنيف الفندق	خمس نجوم			
بداية حجز الفندق	01/01/2012			
نهاية حجز الفندق	06/01/2012			

- الحالة الثالثة:

اسم المعامل	قيمة المعامل	حالة الخدمات		النتيجة
اسم المسافر	أنس	FourSeasons Hotel Reservation	Up	Failure
من مدينة	دمشق	Moskova Hotel Reservation	Up	
إلى مدينة	بيروت	Sheraton Hotel Booking	Up	
تاريخ السفر	01/01/2012	Airline Reservation	Up	
لغاية	06/01/2012	Flight Reservation	Up	
درجة الحجز بالطائرة	درجة رجال الأعمال	الشرح: لم يتم العثور في مستودع الخدمات على خدمة للحجز الفندقي في بيروت ذات تصنيف أربع نجوم.		
تصنيف الفندق	أربع نجوم			
بداية حجز الفندق	01/01/2012			
نهاية حجز الفندق	06/01/2012			

- الحالة الرابعة:

اسم المعامل	قيمة المعامل	حالة الخدمات		النتيجة
اسم المسافر	إياد	FourSeasons Hotel Reservation	Up	
من مدينة	لندن	Moskova Hotel Reservation	Up	☑
إلى مدينة	موسكو	Sheraton Hotel Booking	Up	
تاريخ السفر	01/01/2012	Airline Reservation	Up	
لغاية	06/01/2012	Flight Reservation	Up	☑
درجة الحجز بالطائرة	الدرجة الأولى	الشرح: تم العثور في مستودع الخدمات على خدمة للحجز الفندقي في موسكو ذات تصنيف أربع نجوم.		
تصنيف الفندق	أربع نجوم			
بداية حجز الفندق	01/01/2012			
نهاية حجز الفندق	06/01/2012			

- الحالة الخامسة:

اسم المعامل	قيمة المعامل	حالة الخدمات		النتيجة
اسم المسافر	سامر	FourSeasons Hotel Reservation	Up	☑
من مدينة	اسطنبول	Moskova Hotel Reservation	Up	
إلى مدينة	موسكو	Sheraton Hotel Booking	Up	
تاريخ السفر	01/01/2012	Airline Reservation	Up	
لغاية	06/01/2012	Flight Reservation	Up	☑
درجة الحجز بالطائرة	الدرجة الإقتصادية	الشرح: نلاحظ بأنه لم يتم إجراء الحجز الفندقي في فندق موسكو لأنه مصنّف بأربع نجوم بما يخالف رغبة المستخدم بالحجز في فندق خمس نجوم.		
تصنيف الفندق	خمس نجوم			
بداية حجز الفندق	01/01/2012			
نهاية حجز الفندق	06/01/2012			

- شاشة النتائج:

Trip Planning System

From Location	Damascus	
To Location	Moscow	
Departure Date "dd\mm\yyyy"	01/01/2012	
Your Name	10/01/2012	
Passport ID	123456543	
Flight Class	Business	
Hotel Class	Four Stars	
Hotel Check In Date "dd\mm\yyyy"	01/01/2012	after arrival by 1 hour(s)
Hotel Check out Date "dd\mm\yyyy"	10/01/2012	

Trip Planning　Cancel

Your Trip Info:

Your Information:	
Name	10/01/2012
PassportID	123456543

Ticket Information		**Hotel Information**	
From:	Damascus	Hotel Name:	Moskva 4 Stars Hotel
To:	Moscow	Check In:	01/01/2012 06:00:00 ص
Departure Date:	01/01/2012 03:00:00 ص	Check Out:	10/01/2012 12:00:00 ص
Reservation ID:	98	Reservation ID:	16
Company:	Malaysia Airlines - Tel: 7654	Room ID:	408

Done　Local intranet | Protected Mode: Off　100%

الشكل 7-20، صفحة عرض نتيجة استخدام نظام التخطيط للرحلات الالكتروني

قمنا وبهدف إظهار نتيجة الاستعلام على أنطولوجيات كبيرة وبطريقة عملية بربط نظامنا بأنطولوجية الـ DBPedia[1] والتي تحتوي على 1،9 مليون غرض مصنّف ضمن أنطولوجية للمعلومات المتوافرة على الـWikipedia، حيث يتم الحصول على معلومات إضافية مرتبطة بالأماكن أو الشركات أو الفنادق أو الأشخاص بالربط مباشرةً مع هذه الأنطولوجية، وقمنا بتنفيذ الاستعلامات باستخدام لغة الـ SPARQL[2]، فعلى سبيل المثال يمكن الوصول من خلال صفحة النتائج السابقة إلى مجموعة من المعلومات المتعلقة بمدينة دمشق وبأكثر من طريقة عرض، حيث يعرض الشكل "7-19" معلومات نصية عن المدينة إلى جانب الموقع على الخريطة بينما يعرض الشكل "7-20" مخطط ارتباط الموارد والأغراض المتعلقة بمدينة دمشق.

تجدر الإشارة هنا إلى أن الخدمة الإفتراضية التي تمت إضافتها إلى مستودع الخدمات الدلالية لإتاحة وظيفة التخطيط للرحلات بما فيها من خدمة الحجز الفندقي وحجز رحلات الطيران هي خدمة قابلة لإعادة الاستخدام بشكل مباشر من خلال مكامل الخدمات وهو ما يعزّز مفهوم إعادة استخدام الخدمات المركّبة أيضاً إلى جانب إعادة استتخدام الخدمات الجزئية نفسها، وذلك إلى جانب إمكانية إعادة

1: DBPedia: وهي مجهود عام لاستخراج معلومات مهيكلة من الـWikipedia وجعل هذه المعلومات متاحة على الانترنت بحيث تسمح بطرح استعلامات معقّدة على الـWikipedia, وبالتالي ربط أية معلومات أخرى على الوب بالمعلومات على الـWikipedia. وتحتوي قاعدة المعرفة هذه على أكثر من 3,64 مليون غرض من ضمنها 1,9 مليون غرض مصنّف من أشخاص وأماكن ومنظمات وأفلام وأمراض.

2 SPARQL: وهي اختصار لـ*Protocol and RDF Query Language* , وهي لغة للاستعلام حول عبارات الـRDF والـOWL, وتعتبر من تقنيات الوب الدلالي الأساسية.

استخدامها كخدمة جزئية في أحد سيناريوهات مكاملة خدمات أخرى فعلى سبيل المثال يمكن مكاملة هذه الخدمة مع خدمة جزئية جديدة هي خدمة استئجار سيارة سياحية بالتالي نحصل على خدمة أخرى للتخطيط للرحلات أكثر شمولاً حيث ستتيح بهذه الحالة ثلاث خدمات وهي خدمة الحجز الفندقي وخدمة حجز رحلات الطيران وخدمة استئجار سيارة سياحية.

Information Workbench
WWW PDF Help Login
Damascus
View Blog Revisions

Damascus (دمشق , شام), commonly known in Syria as *Al Sham* (الشام), and as the *City of Jasmine* (مدينة الياسمين), is the capital and the second largest city of Syria and one of the country's 14 governorates. The Damascus Governorate is ruled by a governor appointed by the Minister of Interior. In addition to being one of the oldest continuously inhabited cities in the world, Damascus is a major cultural and religious center of the Levant. The city has an estimated population of 1,711,000 (2009 est.).[1]

Located in southwestern Syria, Damascus is the center of a large metropolitan area of 2.4 million people (2004).[2] Geographically embedded on the eastern foothills of the Anti-Lebanon mountain range 80 km (49.71 mi) inland from the eastern shore of the Mediterranean on a plateau 680 m above sea-level, Damascus experiences a semi-arid climate due to the rain shadow effect. The Barada River flows through Damascus.

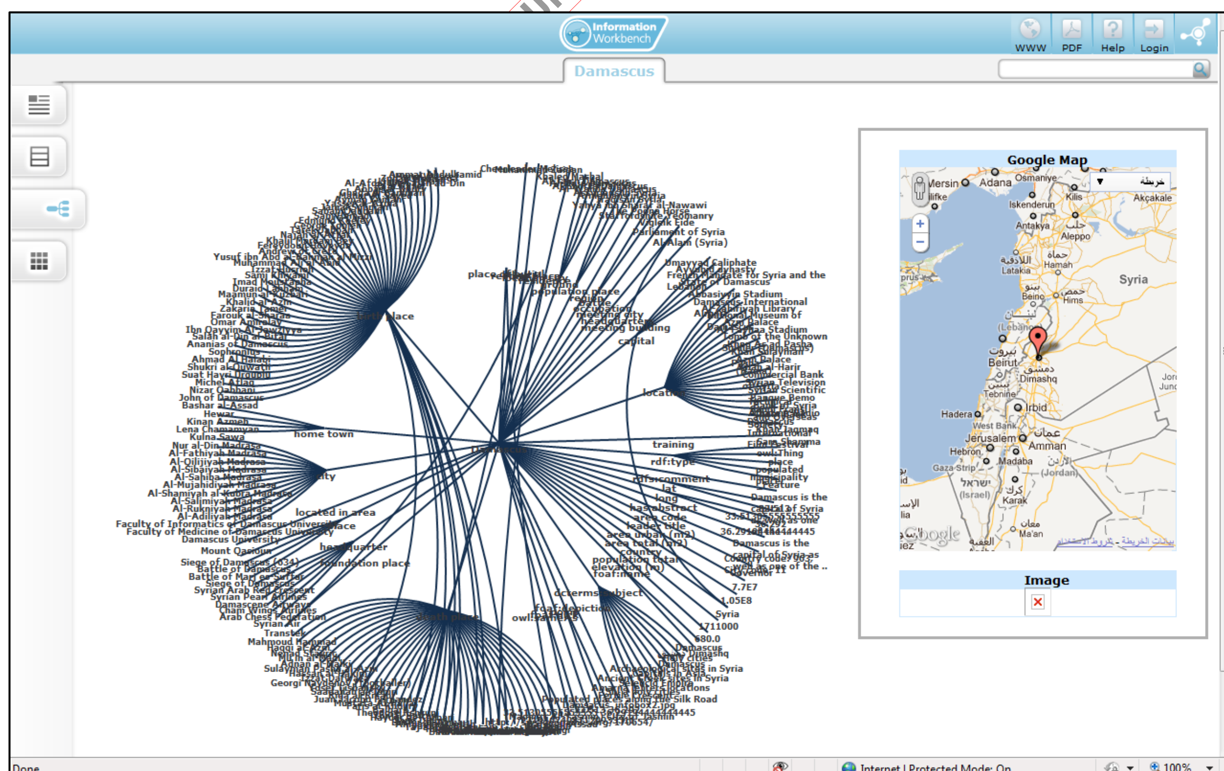
First settled in the 2nd millennium BC, it was chosen as the capital of the Umayyad Caliphate from 661 to 750. After the victory of the Abbasid dynasty, the seat of Islamic power was moved to Baghdad. Damascus saw a political decline throughout the Abbasid era, only to regain significant importance in the Ayyubid and Mamluk periods. During Ottoman rule, the city decayed completely while maintaining a certain cultural prestige. Today, it is the seat of the central government and all of the government ministries. Damascus was chosen as the 2008 Arab Capital of Culture.[3]

Contents

Google Map

Image

الشكل 7-21، معلومات عن مدينة دمشق من خلال قاعدة المعرفة DBpedia



الشكل 7-22، مخطط عن المعلومات المتعلقة بمدينة دمشق من خلال قاعدة المعرفة DBpedia

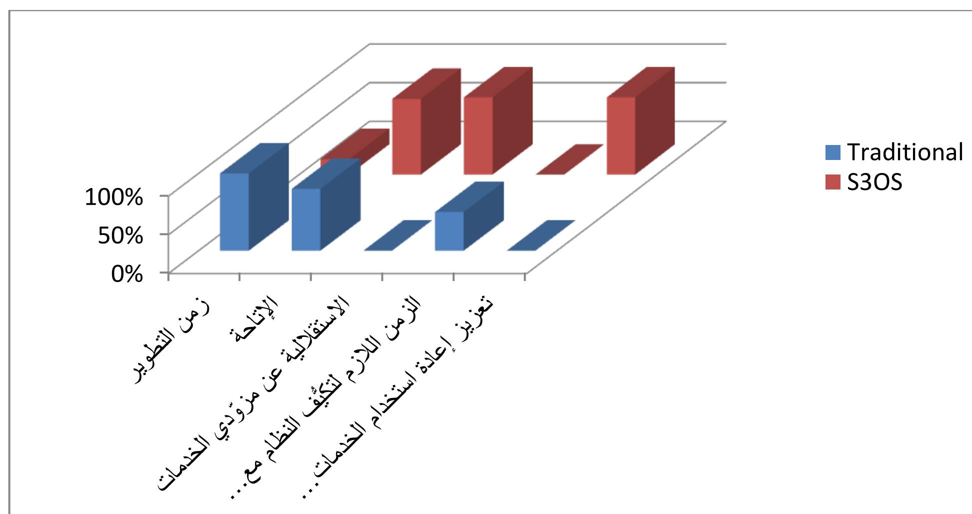
6-7. التقييم:

إن كلاً من الطريقتين السابقتين لبناء نظم المعلومات خدمية التوجّه يتطلّب توافر مجموعة من المتطلّبات الأولية والأساسية لبناء هذه النظم من حيث توافر الخدمات المتعدّدة أما بالنسبة للجانب الدلالي سيتطلّب هذا أيضاً توافر الأنطولوجيات المطلوبة والتوصيف الدلالي المرافق لهذه الخدمات.
انطلاقاً من اعتبار تحقّق هذه المتطلبات أو تساويها في كلا الطريقتين يمكننا تقييم مجموعة النقاط التالية في بناء نظام التخطيط للرحلات الالكتروني من خلال الجدول التالي:

موضوع التقييم	الطريقة التقليدية	الطريقة الدلالية باستخدام إطار العمل المُعتمِد على النموذج S3OS	النتيجة	تفاصيل
احتمالية تلبية متطلبات المستخدم	$\frac{1}{x1 * x2 *..* xn}$ حيث x_i تمثّل عدد خيارات الاستخدام المتاحة في الخدمة i، بالتالي كلما ازداد عدد الخيارات المتاحة تناقص احتمال تلبية رغبات المستخدم. - عدد خيارات تصنيف الفنادق في خدمة الحجز الفندقي=3 - عدد خيارات درجة الحجوزات في خدمة حجز الطيران =3 - عدد خيارات مدن الإنطلاق =10 - عدد خيارات مدن الوصول =10 - احتمال تلبية المتطلبات = 0.11%=1/3*3*10*10	لا ترتبط بشكل مباشر بعدد خيارات الاستخدام وإنما تعتمد على دقة توصيف الخدمات المتوافرة والقدرة على استكشاف هذه الخدمات.		بالرغم من أنه لا يمكن قياس ازدياد احتمالية تلبية متطلبات المستخدمين بدقّة، إلا أن ذلك يبدو جلياً بسبب تقليل عامل التدخل اليدوي في بناء الأنظمة، فمهما كانت خيارات المستخدم متنوعة فستتم تلبيتها في حال توافر خدمات الوب المناسبة، الأمر الذي لا يتوافر بالطريقة التقليدية لِما يتطلبه ذلك من مجهود كبير حتى وإن وُجِدت خدمات الوب تلك.
زمن التطوير	يتطلب البحث عن أكبر عدد ممكن من الخدمات وتصميم طريقة تخاطبها بهدف إتاحة الخدمة الكليّة. - زمن تطوير نظام التخطيط للرحلات الذي تم اختباره = $\frac{1}{2}$ رجل شهر	يتضمّن تصميم سيناريو تكامل مجموعة الخدمات الجزئية المطلوبة وإرفاق الوصف الدلالي ونماذج تفويض الخدمات المطلوبة. - زمن تطوير نظام التخطيط للرحلات الذي تم اختباره = 0.1 رجل شهر	%80	إن استخدام النموذج S3OS سيزيد من إنتاجية تطوير النظم من خلال تقليل الزمن اللازم لبنائها، ففي نظام التخطيط للرحلات انخفض زمن التطوير بمقدار $100 - \frac{0.1}{0.5} = 80\%$
الإتاحة (Availability)	زمن الإتاحة لعينة من 100 خدمة وب في دليل الخدمات SEEKDA = [%98-70]	متوسط زمن الخدمتين الأكثر إتاحة = %98	%28	لأن مدى إتاحة النظام تعتمد على مدى إتاحة الخدمات التي يتألف منها فإن قياسها يعتبر أمراً هاماً. في سبيل ذلك أخذنا عينة من 100 خدمة الوب من موقع البحث عن خدمات الوب

المعيار	الطريقة التقليدية	S3OS	التحسين	التعليل
				SEEKDA والذي يزوّدنا بمدى إتاحة الخدمات.
الاستقلالية عن مزوّدي الخدمات	%0	%100	%100	إن بناء النظام بالطريقة التقليدية يعتمد بشكل كامل على الخدمات ومزوّديها بسبب الإعتماد المباشر على خدمات محدّدة العنوان. أما باستخدام البعد الدلالي فالنظام لا يعتمد بالأساس على مزوّدي خدمات محدّدين وإنما قد يتم الإعتماد على مزوّدي خدمات مختلفين في كل مرة يجري فيها استخدام النظام.
الزمن اللازم لتكيُّف النظام مع التغييرات على الخدمات التي يتألف منها	0- 15 يوم	0 يوم	0 – %100	لأن النظام الدلالي لا يعتمد على خدمات محدّدة فإن أي تغيير يطرأ على خدمات الوب لن يؤثر عليه، في الوقت الذي قد يتطلب ذلك تعديل بسيط أو تعديل كامل على النظام التقليدي.
تعزيز إعادة استخدام الخدمات الجديدة	%0	%100	%100	إن النموذج S_3OS يعتمد على بناء الوظيفة الجديدة المطلوبة في النظام كخدمة مركّبة يمكن التعامل معها كأية خدمة بسيطة مما يتيح إعادة استخدامها بشكل كامل الأمر الغير متوافر في الطريقة التقليدية.

يوضّح الشكل "7-23" مخطط بياني بالمقارنة بين الطريقتين:



الشكل 7-23، تقييم طريقتي بناء التطبيق

7-7. خلاصة:

بيّنا من خلال التطبيق العملي في هذا الفصل أهم النقاط والخطوات لبناء أنظمة تعتمد في بنيتها على خدمات الوب بالطريقتين التقليدية والدلالية بحيث تم إيضاح الإختلافات الأساسية بين الطريقتين وما يضيفه نموذجنا على آلية بناء وإعادة استخدام وتركيب الخدمات دلالياً. حيث قمنا بتنفيذ مجموعة من حالات الاستخدام باستخدام مكامل الخدمات الذي طوّرناه لهذا الغرض وذلك للوقوف على خطوات وآلية اختيار الخدمات وتنفيذها.

خلاصة وآفاق

يساهم هذا البحث في تعزيز استخدام خدمات الوب كمكوّن أساسي في نظم المعلومات خدمية التوجّه، إلى جانب توفير مرونة أكبر في بناء هذه النظم وإدارتها من خلال تأمين آلية ديناميكية لتقليل التدخل اليدوي بعد بناء هذه النظم والذي قد يكون ضرورياً في حالة الخلل في أداء بعض الخدمات لدورها أو في حال وجود أي تغيير في بنية الخدمة نفسها أو في طريقة استخدامها، كما وتسمح هذه الطريقة بتقليل الجهد اللازم لبناء هذه النظم من خلال القدرة على البحث الآلي عن الخدمات بشرط توافرها إلى جانب وصفها الدلالي في مستودع الخدمات. يساهم النموذج الذي قدمناه بـ:

- توفير الزمن اللازم لبناء نظم المعلومات خدمية التوجّه.
- زيادة موثوقية الأنظمة خدمية التوجّه من خلال تمكينها من إيجاد البدائل عند مواجهة أية مشاكل.
- استقلالية النظم خدمية التوجّه عن مزوّدي الخدمات.
- تقليل زمن إدارة وتعديل هذه النظم.
- تعزيز إعادة استخدام خدمات الوب.
- إمكانية إعتبار هذه النظم كخدمات يجري إعادة استخدامها أيضاً.
- تقليل حجم التدخل البرمجي.
- زيادة فرصة تلبية احتياجات المستخدم من خلال وظيفة البحث والمطابقة والتنفيذ بما يتلائم مع رغبات المستخدم.

اعتمد النموذج S_3OS المُقترح على الأنطولوجية OWL-S والتي تقدِّم نموذجاً لخدمات الوب الدلالية. تم ربط الخدمات بعضها مع بعض من خلال وصفها الدلالي لجعلها أكثر ديناميكية حيث تم تحويل الخدمة نفسها إلى طالب لخدمة مكافئة لها بالمواصفات وذلك من خلال إطار عمل عام يوفِّر الوظائف الضرورية لذلك من نشرٍ للخدمات وتوصيفها ومكاملتها وتنفيذها.

قمنا بهدف اختبار إطار العمل بإجراء التحقيق البرمجي لمكوّناته والتي تُمكّن مزوّدي الخدمات من نشر خدماتهم وتوصيفها دلالياً وتعريف وظائف جديدة قابلة لإعادة الاستخدام بالإعتماد على الخدمات التي يُقدِّمونها، كما ويتيح لمستهلكي الخدمات من استخدام مكامل الخدمات في تنفيذ سيناريوهات المكاملة الموجودة. قمنا كتطبيق عملي ببناء نظام للتخطيط للرحلات باستخدام إطار العمل السابق وأجرينا إختبار لمجموعة من حالات الإستخدام لتقييم مجموعة من العوامل المؤثّرة في عملية بناء النظم وكيفية تحسُّنها من احتمالية تلبية رغبات المستخدم النهائي، والزمن اللازم للتطوير، والإستقلالية عن مزوّدي الخدمات، ومدى إتاحة النظام، ومقدار التدخل البرمجي اللازم لصيانة وتعديل النظام وغيرها من العوامل. قمنا

وبهدف إظهار نتيجة الاستعلام على أنطولوجيات كبيرة وبطريقة عملية بربط النظام بأنطولوجية الـDBPedia والتي تحتوي على 9،1 مليون غرض مُصنّف ضمن أنطولوجية للمعلومات المتوافرة على الـWikipedia.

تجدر الإشارة هنا إلى أن نجاح عمل نموذجنا أو أي نموذج مشابه يعتمد بشكل كبير على حجم الأنطولوجيات المتوافرة وتنوعها لأنها تعتبر أحد المكونات الأساسية في الوب الدلالي، ذلك إلى جانب حجم مستودع الخدمات الدلالية ومدى توافر خدمات الوب الموصّفة دلالياً، حيث تزداد كفاءة النموذج كلما احتوى مستودع الخدمات على قدرٍ أكبر من خدمات الوب مما يتيح وجود بدائل دائمة للخدمات بعضها عن بعض، كما ويعتمد على مدى تنوع هذه الخدمات لتشمل مجالات عمل متعدّدة تلبي احتياجات أنظمة متعدّدة الأغراض.

يُمكن العمل مستقبلاً على إجراء مجموعة من التطويرات والأبحاث ومنها:

1- تطوير آلية تُلغي أي نوع من التدخل اليدوي لبناء الأنظمة خدمية التوجّه وذلك من خلال إمكانية الربط بين الخدمات بطريقة آلية بحيث يكون مكامل الخدمات قادراً وبشكل آلي على معرفة الخدمات التي يتألف منها النظام المطلوب وتحديد آلية تخاطبها بدون تحديد الخدمات، كأن نكتفي بالقول بأن نظام التخطيط للرحلات يتألف من خدمة الحجز الفندقي وحجز رحلات الطيران.

2- تطوير إطار العمل المقترح بحيث تُمثّل شبكة الوب مستودع الخدمات وما يتطلبه ذلك من محركات بحث عن خدمات الوب الدلالية، وعندها يفترض اشتراط نشر الوصف الدلالي عند نشر أي خدمة وب.

3- تطوير النموذج بحيث يتيح مجموعة من الخيارات للمستخدم النهائي ضمن أوزان معيّنة من خلال إضافة آلية للتعلم الذاتي.

نعتقد بأن هذه الأبحاث ستواجه عدداً من التحديات كقابلية التوسّعة وموثوقية خدمات الوب نفسها ومدى الثقة بين الخدمات المرتبطة لتأمين خدمات جديدة بعضها مع بعض.

المراجع

[Aalst 2008] W.M.P. van der Aalst and H.M.W. Verbeek. Process Mining in Web Services. Eindhoven University of Technology. Sep 2008.

[BARNABY 2002] Tom Baranaby. Distriduted Programming. Jun 2002.

[Barry 2003] Tom Barry. Web Services and Service-Oriented Architecture. Morgan Kaufmann Publishers. 2003.

[Barry 2008] Douglas K. Barry. Web Services and Service-Oriented Architecture: The Savvy Manager's Guide, ISBN:1558609067. May 2008.

[Beale, Heard, Kalra, Lloyd 2005] T. Beale, S. Heard, D. Kalra, and D. Lloyd. The openEHR Data Structures Information Model, Revision 1.6rc1 (Release 1.0 draft). openEHR Reference Model, the openEHR foundation, 2005.

[Beale, Heard 2005] T. Beale and S. Heard. Archetype Definition Language (ADL), Revision 2.0rc1 (Release 1.0 draft). openEHR Specification, the openEHR foundation, 2005.

[Berners-Lee 2001] Berners-Lee, Tim. The Semantic Web. Scientific American Magazine. 2001.

[Bicer, Laleci, Dogac, Kabak 2005] V. Bicer, G. Laleci, A. Dogac, and Y. Kabak. Artemis Message Exchange Framework: Semantic Interoperability of Exchanged Messages in the Healthcare Domain. ACM Sigmod Record, 34(3), September 2005.

[Boniface, Wilken 2005] M. Boniface and P. Wilken. ARTEMIS: Towards a Secure Interoperability Infrastructure for Healthcare Information Systems. In T. Solomonides, R. McClatchey, V. Breton, Y. Legr´e, and S. Nørager, editors, Studies in Health Technology and Informatics – From Grid to Healthgrid: Proceedings of Healthgrid 2005, volume 112, pages 181–189. IOS Press, April 2005.

[Brickley, Guha 2004] D. Brickley and R.V. Guha. RDF Vocabulary Description Language – RDF Schema. http://www.w3.org/TR/rdf-schema/, 2004.

[Brien, Marakas 2007] James O'Brien, George Marakas. Introduction to Information Systems, McGraw-Hill/Irwin; 13 edition, ISBN 0073043559. 2007.

[Chu 2004] Robert Chu. Web Services Standards for Data Mining. Aug 2004.

[Cobo, Losada, Benjamins, Nino, Contreras 2004] J.M. Lopez-Cobo, S. Losada, O. Corcho, R. Benjamins, M. Nino and J. Contreras. Semantic Web Services for Financial Overdrawn Alerting. Proceedings of the 3rd International Semantic Web Conference (ISWC-2004), 2004.

[Craig 1998] E. Craig. Ontology. In E. Craig, editor, Routledge Encyclopedia of Philosophy, pages 117–118. Routledge, New York, 1998.

[DAML 2008] DAML Servies. http://www.daml.org/services/owl-s. 2008.

[Dignum 2003] Frank Dignum, Virginia Dignum, Mehdi Dastani. Software Agents Theory and Practice. Utrecht University. Sep 2003.

[Esmaeil 2003] Bilal Esmaeil. Scripting Languages and software components. Albaath University. Aug 2003.

[Fensel 2004] D. Fensel. Intelligent Web Services. Leopold-Franzens-Universität Innsbruck. 2004.

[Fujii 2006] Keita Fujii, Tatsuya Suda. Web Service Composition. Mar 2006.

[GOMBOTZ 2005] ROBERT GOMBOTZ, KARIM BAÏNA, and SCHAHRAM DUSTDAR. Towards Web Services Interaction Mining Architecture. Vienna University of Technology. 2005.

[Gomez-Perez 2002] Gomez-Perez, Asuncion. Ontology Specification Languages for the Semantic Web. 2002.

[Gruber 1993] T.R. Gruber. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 6(2):199–221, 1993.

[Guarino 1998] N. Guarino. Formal Ontology and Information Systems. 1998.

[Harmelen, Huang, Stuckenschmidt, Sure 2005] F. van Harmelen, Z. Huang, H. Stuckenschmidt, and Y. Sure. A Framework for Handling Inconsistency in Changing Ontologies. 2005.

[Heflin 2001] Heflin, J & Hendler, J. A Portrait of the Semantic Web in Action. 2001.

[Heflin 2001] Heflin, J. Towards the Semantic Web: Knowledge Representation in a Dynamic, Distributed Environment. 2001.

[Heijst 1997] G. van Heijst, A.Th. Schreiber, and B.J Wielinga. Using explicit ontologies in kbs development. International Journal of Human Computer Studies, 45:148-292, 1997.

[Jakoniene 2003] Vaida Jakoniene. Ontology Integration, 2003.

[Klyne, Carroll 2004] G. Klyne and J. Carroll. RDF Concepts and Abstract Syntax. http://www.w3.org/ TR/rdf-primer/, 2004.

[Korotkiy 2007] Maksyn Korotkiy, Jan Top. Reference Ontology for Semantic Service Oriented Architectures. Vrije Universiteit Amsterdam. May 2007.

[Krogstie 2009] John Krogstie, Csaba Veres and Guttorm Sindre. Interoperability in Semantic Web Services. 2009

[Lord 2009] Phillip Lord, Sean Bechhofer, Mark D. Wilkinson, Gary Schiltz, Damian Gessler, Duncan Hull, Carole Goble, and Lincoln Stein. Applying Semantic Web Services to Bioinformatics. Jan 2009.

[Maier 2003] Andreas Maier, J. Aguado, A. Bernaras, Integration with Ontologies. Conference Paper WM2003, April 2003, Luzern.6

[McIlraith 2002] ch5 [3] S. McIlraith and T.C. Son. Adapting Golog for Composition of Semantic Web Services. Proceedings 8th International Conference on Knowledge Representation and Reasoning, 2002.

[McIlraith, Martin 2003] S. McIlraith and D. Martin. Bringing Semantics to Web Services. IEEE Intelligent Systems. 2003.

[Microsoft 2006] Enabling “Real World SOA” through the Microsoft Platform Microsoft Corporation. December 2006.

[Microsoft 2006] Learn About Service Oriented Architecture (SOA), December 2006.

[Motik, Sattler 2006] B. Motik and U. Sattler. A Comparison of Reasoning Techniques for Querying Large Description Logic ABoxes. 2006.

[Motik, Sattler, Studer 2004] B. Motik, U. Sattler, and R. Studer. Query Answering for OWL-DL with Rules. 2004.

[Niles, Pease 2001] I. Niles and A. Pease. Towards a Standard Upper Ontology. 2001.

[Paolucci, Sycara 2003] M. Paolucci and K. Sycara. Autonomous Semantic Web Services. IEEE Internet Computing.September 2003.

[Parallab 2004] ch1 [1] Software Reusability and Efficiency undertaken by Parallab, Bergen Center for Computational Science, University of Bergen (Norway) for the ENACTS network. APRIL 2004.

[Pepper, Moore] S. Pepper and G. Moore. XML Topic Maps (XTM) 1.0. http://www.topicmaps. org/xtm/1.0/.

[Preist 2004] C. Preist. A Conceptual Architecture for Semantic Web Services. Proceedings of the 3rd International Semantic Web Conference (ISWC-2004), 2004.

[Reiter 1980] R. Reiter. A Logic for Default Reasoning. Artificial Intelligence, 13:81–132, 1980.

[Russel, Norvig 1995] S. Russel and P. Norvig. Artificial Intelligence – A Modern Approach. Prentice-Hall, 1995.

[Schneider, Hayes, Horrocks 2002] P.F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language; Semantics and Abstract Syntax. http://www.w3.org/TR/owl-semantics/, November 2002.

[Sirin, Parsia, Grau, Kalyanpur, Pellet 2005] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A Practical OWL DL Reasoner. Technical report, University of Maryland Institute for Advanced Computer Studies (UMIACS), 2005. http://mindswap.org/papers/PelletDemo.pdf.

[Sowa 2000] J.F. Sowa. Knowledge Representation. Brooks Cole Publishing, Pacific Grove, CA, USA, 2000.

[Stollberg 2005] Michael Stollberg and Armin Haller. Semantic Web Services Tutorial. DERI – Digital Enterprise Research Institute. 2005.

[Sure, Erdmann, Angele, Staab, Studer, Wenke 2002] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. OntoEdit: Collaborative Ontology Development for the Semantic Web. In ISWC '02: Proceedings of the First International Semantic Web Conference on The Semantic Web, pages 221–235. Springer-Verlag, 2002.

[Tolle 2001] Kristin Michele Tolle, Hsinchun Chen. Intelligent Software Agents for Electronic Commerce. University of Arizona. Sep 2001.

[Trastour, Bartolini, Preist 2003] D. Trastour, C. Bartolini and C. Preist. Semantic Web Support for the B2B E-Commerce Pre-Contractual Lifecycle Computer Networks. 2003.

[Ullman 1989] Jeffrey D. Ullman. Principles of Database and Knowledge-Base Systems: Volumes I and II. Computer Science Press, 1989.

[Uschold 1996] M. Uschold and M. Gruninger. Ontologies: principles, methods, and applications. Knowledge Engineering Review, 11(2):93-155, 1996.

[Valle, Cerizza 2005] E. Della Valle and D. Cerizza. The mediators centric approach to automatic web service discovery of glue. In M. Hepp, A. Polleres, F. van Harmelen, and M.R. Genesereth, editors, MEDIATE2005, volume 168 of CEUR Workshop Proceedings, pages 35–50. CEUR-WS.org, 2005. online http://CEUR-WS.org/Vol-168/MEDIATE2005-paper3.pdf.

[Vturk 2007] Vturk. Enabling Real-World SOA to connect your business vision with software. Jul 2007.

[W3C 2001] World Wide Web Consortium (W3C). W3C Semantic Web Frequently Asked Questions. 2001.

[Yang, Kifer, Zhao 2003] G. Yang, M. Kifer, and C. Zhao. Flora-2: A Rule-Based Knowledge Representation and Inference Infrastructure for the Semantic Web. 2003.

[Yue, Zuoquan 2006] M. Yue and L. Zuoquan. Infering with Inconsistent OWL DL Ontology: a Multi-valued Approach. 2006.